# Systematic Asset Management

Tung-Lam Dao[1]

[1]Independent Researcher, Paris

VNFinance Conference, 23th June 2018

**Abstract**

Since the last decade, financial technology (Fintech) has made a lot progresses in many angles of the finance industry from the novel concepts of the transaction to the systematic/intelligent management of financial products. Back to the 80s, the first attempts to combine applied mathematics, numerical algorithms with high-performance computers in trading and portfolio construction gave birth to a new trend of asset management "systematic asset management". Employing computers to perform complex calculations, to estimate the optimal trading quantities have improved the gain probability and the risk management. Systematic asset management must be considered as one of the first revolutions in financial technology. However, it quickly became the industrial secret of many successful hedge funds such as Renaissance, D.E.Shaw, Two Sigmas, CFM, e.t.c. The 2008 crisis has changed the investment point of view of investors and the regulators. They required more and more efforts from the hedge fund industry and asset management in the transparency of their portfolios and their risk management. Some management styles such as "Smart beta" or "Risk Parity" were revealed to the large public with very detailed explanation both on the concept and the implementation. Recently, a new class of investment strategies named "alternative beta" or "alternative risk premium" was also opened to the public. It consists of combining well-known trading strategies with systematic risk management in order to offer high performance and decorrelated return to the market benchmark. These last evolutions allowed more and more opportunities for Fintech to improve the systematic asset management. Machine learning and AI can be employed to explore novel trading strategies, to detect anomal risks, to reduce the operational risks or to simulate stress-scenarios whereas blockchain is a great candidate for future improvement of the current transaction system. The objective of this note is to explain the main principles of the systematic asset management through some simple examples. We expect that our approach may be useful to identify the potential applications of Fintech in this domain.

## 1 Introduction

### 1.1 What is the definition of "*Fintech*"?

- Fintech definition: "Financial technology" Wikipedia
- Fintech landscape Investopedia
- More detail on the list of technologies The Fintech 50: The Complete List
- Learn and hear about it every day through the media, the social network (facebook, linkedin), e.t.c

### 1.2 What is the place of "*Fintech*" in Asset Management?

- Wikipedia: "*Financial technology has been used to automate insurance, trading, and risk management*".
- Investopedia: "*Robo-advisors, such as Betterment, utilize algorithms to automate investment advice to lower its cost and increase accessibility*".
- The Fintech 50: The Complete List: Many start-ups on robo-advisors, machine learning/AI on alternative data, e.t.c

### 1.3 Quantitative/Systematic Asset Management

The intensive use of applied mathematics and computers in quantitative trading has changed the *Asset Management* considerably since the last three decades:

- Employing computers to perform complex calculations, to estimate the optimal allocation and trading made a lot of improvements and gave birth to a great number of quantitative hedge funds such as Renaissance, D.E.Shaw, Two Sigmas, CFM, e.t.c
- Optimization of Portfolio and Systematic Risk Management aims to raise significant *asset under management* for hedge funds such as Bridge Water, AQR, e.t.c

- With operational risks in banks and asset management, employing new technologies with high performant computers help to eliminate the potential loss due to bad risk management by simulating "stress-test" scenarios.
- For long time, statistical methods and filter techniques are main tools to seek "trading strategies". Machine learning and AI have shown a great power in many domains and will potentially take it role in quantitative finance.

### 1.4 Outline

- Quick introduction to quantitative/systematic asset management.
- Main ideas of systematic management through some simple examples.
  - Asset modeling
  - Simple idea of systematic trading strategies
  - Simple construction of portfolio
  - Risk management
- Conclusions and perspectives

## 2 Systematic Management

In this section, we explain the main principle of a systematic decision in asset management

### 2.1 Systematic decision

- Systematic decision requires the porfolio managers to be confident on the implementation of the portfolios.
  - Systematic both in the bull and bear markets
  - Systematic on the capital allocation between products
  - No performance driven/ No human intervention
- The decision process of the trading systematic concerns three main parts:
  - Finding trading signals for an universe of financial instruments
  - Combing trading signals to build a portfolio

  - Managing the risk of the instruments and of the portfolio
- Remarks:
  - Another important part is the systematic execution.
  - This part is usually masked by the "high frequency" trading.

## 2.2 Suggestion of literature

- The basis of quantitative asset management and risk management is discussed in the quantitative finance literature by many authors.
- However, I recommend here to use some important results from:
  - Two books of *Thierry Roncalli* on Quantitative Management[1] and Risk Management contain many detailed and useful implementations[2].
  - Paul Wilmott's book on Quantitative Finance [3]
  - Jean-Philippe's book on Risk Financial and derivatives [4]

## 2.3 Finding trading strategies "*Alpha*"

- There are many ways to build a systematic trading strategy, the most common way is based on:
  - Collecting relevant data to the market and the asset prices
  - Learning from data the predictive patterns that may be relevent to predict the future movement of asset prices.
- For the data, we need first to clean up the data in respecting the causality criteria. Example of data:
  - standard data such as price, volume, order books from high frequency to monthly or yearly timescale.
  - non-standard data such as weather, satellite, trade flow, retail, internet, social network data.
- For the technique, we can profit from important toolkits in other domain such as linear filters, non-linear filters, statistical learning, AI.

### 2.3.1 How to judge the prediction quality?

Predicting the future price of the assets is the most difficult part of the systematic management.

- It's very frustrated. For a Normal distribution of return, if we can predict with:
  - probability of 55%, it's already a good predictor (SR=2).
  - probability of 60%, you will be hired in the best hedge funds (SR=4).
  - average prediction for a single product is around 51% (SR=0.4).

```
In [8]: # Basic imports
        import pandas as pd
        import numpy as np
        import seaborn as sns
        import cvxpy as cvx
        from scipy.stats import multivariate_normal
        sns.set(color_codes=True)
        %pylab inline
```

```
Populating the interactive namespace from numpy and matplotlib
```

```
In [9]: # Prediction probability with normal distribution
        from scipy.stats import norm
        sr = [2, 4, 0.4]
        rv = norm()
        prb = [rv.cdf(s * 1 / sqrt(252)) for s in sr]
        df = pd.DataFrame(index=['Good', 'Best HF', 'Average'])
        df['Sharpe ratio'] = sr
        df['Probability (%)'] = 100 * np.round(prb, 2)
        df
```

Out[9]:

| | Sharpe ratio | Probability (%) |
|---|---|---|
| Good | 2.0 | 55.0 |
| Best HF | 4.0 | 60.0 |
| Average | 0.4 | 51.0 |

- How to judge a predictor?
  - realize long backtest (depending on investment horizon)
  - use statistic tests such as: sharpe ratio ($\mu/\sigma$), t-stats, information ratio, etc
  - compare to convetional benchmarks: market performance, CTA, risk-parity, hedge-fund index

### 2.3.2 How to improve the prediction results?

- Here are some suggestions to improve the prediction:
  - Get a better understanding of the asset price process
  - Use the right filters to build prediction
  - "Large number law":
    - Apply the same strategy for many decorrelated assets to profit from diversification
    - Apply more frequently in order to increase the statistics
- **Warning**: Always be awared of "future information" and "over fitting"

## 2.4 Building portfolios "*Beta*"

- Once trading ideas are synthesized in the predictive signals, one may ask the following questions:
  - what is the optimal way to combine the different signals?
  - what is the optimal way to trade a portfolio of assets?
- It concerns mainly the optimal construction of the portfolio and has a very long history back to the time of Markowitz and the theory of Sharpe.
- Two common approaches are:
  - Passive allocation: risk-based methods
  - Dynamic allocation: performance-driven methods

### 2.4.1 Example: gain vs cost optiomization

- The main idea is to optimize the gain against the cost and under certain risk and exposure constraints.

$$\begin{aligned} \max \quad & Gain - Cost \\ \text{u.c.} \quad & Risk,\ Exposure, e.t.c \end{aligned}$$

- Remarks:
  - This is a very intuitive and common approach.
  - The use of constraints reflect the vision of the portfolio managers about the market.
  - "The evil is in the detail": parameter estimation, appropriate constraints are the main ingredients for the success.

### 2.4.2 Remarks on portfolio construction

- This general problem is usually solved by an optimization program using convex optimization algorithm. The two main difficulties of this optimization program are:
  - The estimation of parameters: performance, costs, risks, e.t.c
  - The computation performance at high dimensionality
- Example: considering a portfolio of 500 assets using 20 trading strategies, we may have $10^4$ individual timeseries of daily performance. We need then to estimate all the parameters of $10^4$ timeseries, then build an optimization for a vector of variables $\mathbf{w}$ of dimension $10^4$ (optimal weights for each couple asset/strategy). Running a daily backtest over 20 years may require $5 \times 10^3$ optimizations.

## 2.5 Risk management

- An important part of the asset management is the risk management.
- This process concerns many important questions:
  - How to measure or to monitor the risks?
  - How to model the crash scenario or stress-test?
  - How to control and manage the risks?
  - How to allocate capital based on risk constraints?

# 3 Systematic Asset Management via examples

We present some examples with *simulated data* of asset prices:

- Prices are modeled by geometric Gaussian distributions with long-term/short-term auto-correlation.
- Use the filter as a tool for building predictors.
- Use simple risk measurement to monitor the portfolio.
- We backtest portfolios with different allocation scheme:
  - Passive portfolio: risk-based construction
  - Dynamic portfolio: performance-based construction

## 3.1 Modeling asset prices

- Well-known model for asset price is the famous Black-Scholes:

$$\frac{dS_t}{S_t} = \underbrace{\mu_t}_{\text{trend}} \times dt + \underbrace{\sigma_t}_{\text{volatility}} \times \underbrace{dB_t}_{\text{noise}} \tag{1}$$

  where $\mu_t$ is the long-term trend, $\sigma_t$ is the daily volatility and $B_t$ is a Brownian process.

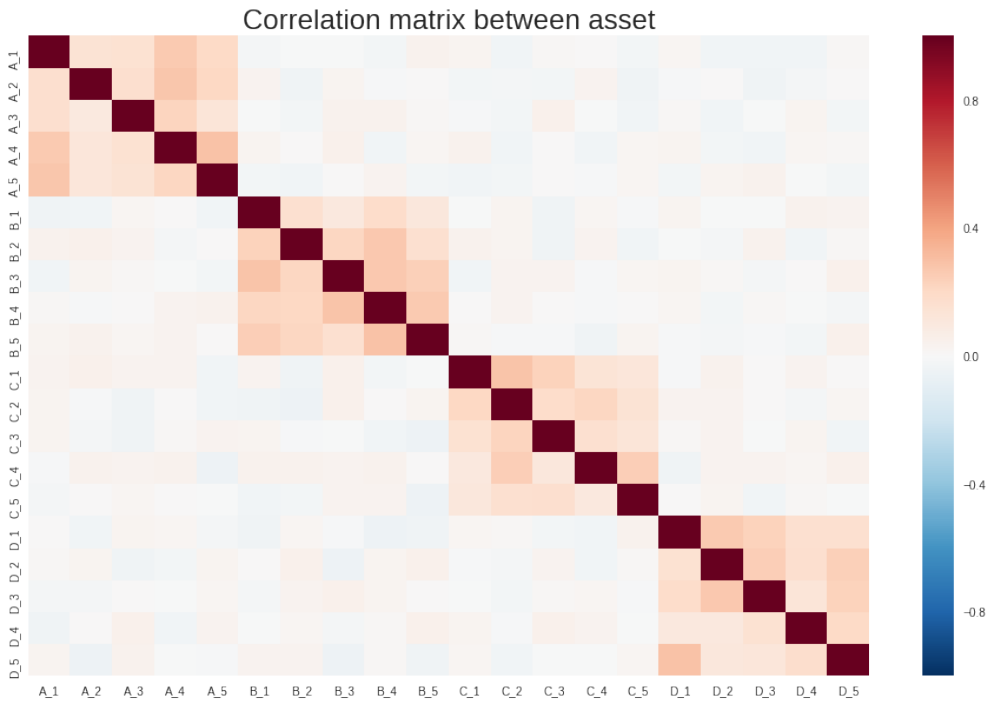
- In order to introduce the "alpha" into this price process, we can introduce:
  - positive drift $\mu_t$
  - short-term auto-correlation noise instead of a Brownian process

### 3.1.1 Simple implementation

In [10]:
```
# Drift and Correlation matrix
n_assets = 20
begin_date = datetime.datetime(1996, 1, 1)
end_date = datetime.datetime(2018, 1, 1)
date_ranges = pd.date_range(begin_date, end_date, freq='B')
n_days = len(date_ranges)
year = 252.
sigma = 0.1 / sqrt(year)
sect_asset = ['A', 'B', 'C', 'D']
n_sect = len(sect_asset)
name_assets = [sect + '_' + str(n + 1) for n in
               range(n_assets / n_sect) for sect in sect_asset]
ref_assets = {sect + '_' + str(n + 1): 'Sector_' + sect  for n
              in range(n_assets / n_sect) for sect in sect_asset}
mu = pd.Series(0.05 * rand(n_assets) / year, index=name_assets)
prd_rho = 0.2
sect_rho = 0.1
cov_mat = dict()
for i in name_assets:
    cov_mat[i] = dict()
    for j in name_assets:
        if i==j:
            cov_mat[i][j] = 1.
        elif i[0]==j[0]:
            cov_mat[i][j] = prd_rho + 0.2 * (rand() - 0.5)
        else:
            cov_mat[i][j] = sect_rho * (rand() - 0.5)
cov_mat = pd.DataFrame(cov_mat)
```

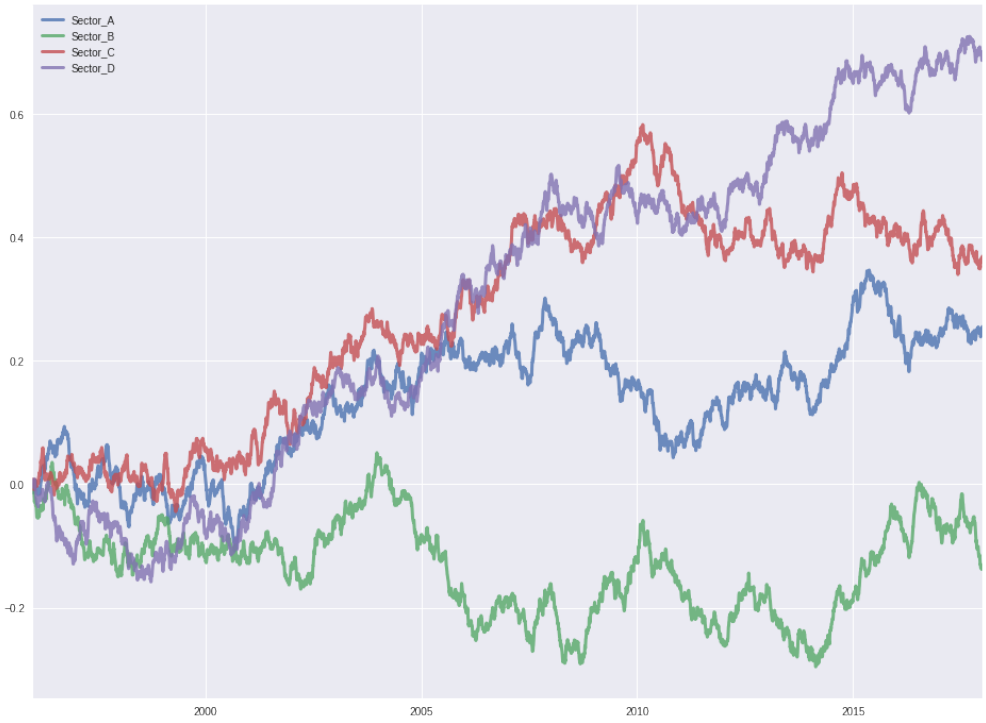
In [11]:
```
# Asset price modeling
rvs = multivariate_normal(np.zeros(n_assets), cov_mat, n_days)
dB = pd.DataFrame(rvs, columns=name_assets, index=date_ranges)
dB = dB.ewm(0.5).mean()
dB /= dB.std()
ret = mu + sigma * dB
```

In [12]:
```
#
f = figure(figsize=(16, 10))
ax = f.add_subplot(111)
sns.heatmap(cov_mat, ax=ax)
title('Correlation matrix between asset', fontsize=24);
```




### 3.1.2 Data generation

```
In [13]: # Simulated data of asset price
         f = figure(figsize=(16, 12))
         ax = f.add_subplot(111)
         sect_ret = ret.groupby(lambda x: ref_assets[x], axis=1).mean()
         sect_ret.cumsum().plot(ax=ax, lw=3, alpha=0.8);
```




## 3.2 Momentum strategy

- We consider a traditional class of strategy "*trend following*"[5, 6, 7] (for more detail)
- The main idea is to predict the tomorrow price based on recent prices:
  - If the recent prices are in the increasing trend, we buy assets to profit the gain
  - Otherwise, we sell assets to stop the loss
- To dectect the recent trend in a time series $y_t$, we decompose it in two components: slowly varying trend and a rapidly varying noises:

$$y_t = x_t + \varepsilon_t$$

  - $x_t$ is the hidden trend (*signal*)
  - $\varepsilon_t$ is the random variation in the price due to market movement or trading (*noise*)

### 3.2.1 Simple prediction: EWMA filter

- A trivial method is using the moving average prediction. For a practical reason, the "Exponenital Weighted Moving Average" is usually employed [9]:

$$x_t = Ema_T(y_t) = \sum_i w_{t-i} y_{t-i} \tag{2}$$

- Simple iterative implementation

$$x_t = \frac{1}{T} y_t + \left(1 - \frac{1}{T}\right) x_{t-1} \tag{3}$$

- only one simple parameter $T$ the timescale

- Directly related to a simple version of Kalman filter
- Empirically works for many products where the future prices are related to recent past.

*Other examples of filter for momentum strategies*[8]

### 3.2.2 More sophisticated method: L2 filter

$L_2$ filter (so-called Hodrick-Prescott filter)[10] consists to determine the trend $x_t$ by minimizing:

$$\frac{1}{2}\sum_{t=1}^{n}(y_t - x_t)^2 + \lambda \sum_{t=2}^{n-1}(x_{t-1} - 2x_t + x_{t+1})^2$$

The Hodrick-Prescott scheme can be rewritten in the vectorial space $\mathbb{R}^n$ and its $L_2$ norm $\|\cdot\|_2$ as:

$$\frac{1}{2}\|y - x\|_2^2 + \lambda \|Dx\|_2^2$$

$$D = \begin{bmatrix} 1 & -2 & 1 & & & & \\ & 1 & -2 & 1 & & & \\ & & & \ddots & & & \\ & & & 1 & -2 & 1 & \\ & & & & 1 & 2 & 1 \end{bmatrix} \tag{4}$$

The exact solution of this estimation is given by

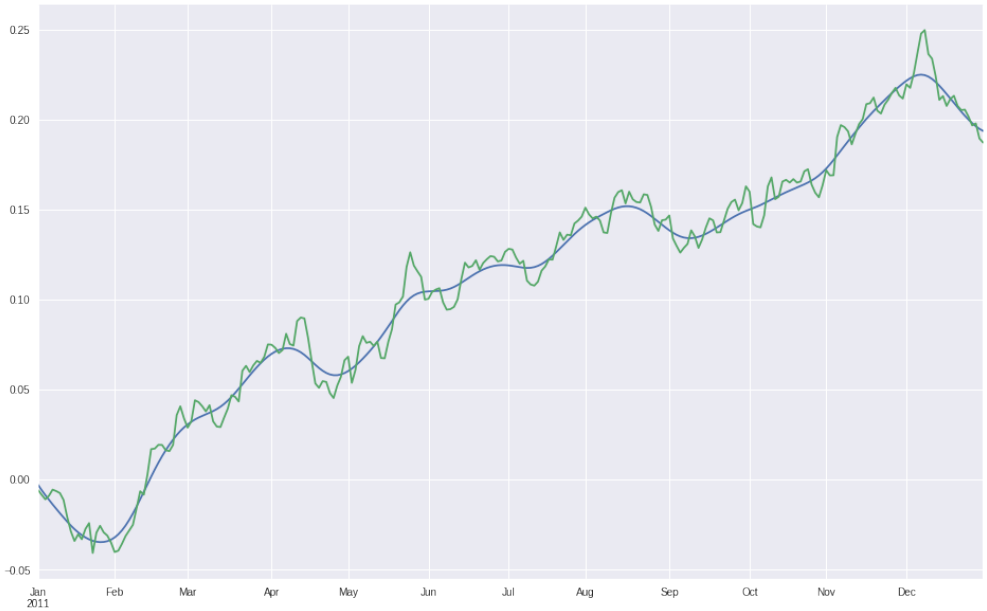
$$x^\star = \left(I + 2\lambda D^\top D\right)^{-1} y$$

### 3.2.3 Implementation

```
In [14]: # Implementation of L2 filter
         def build_l2_matrix(n_sample, lbd):
             """ Build L2 matrix
             """
             D = np.zeros((n_sample - 2,  n_sample))
             for i in range(n_sample - 2):
                 for j in range(1, n_sample):
                     if j==i:
                         D[i,j] = 1
                     if j==i+1:
                         D[i, j] = -2
                     if j==i+2:
                         D[i, j] = 1
             I = np.diag(np.ones(n_sample))
             L = (I + 2 * lbd * np.dot(D.T, D))
             return L

         def l2_filter(ret_sample, L):
             """ Simple implementation of L2 filter
             """
             cols = ret_sample.columns
             idxs = ret_sample.index
             trend = np.dot(np.linalg.inv(L), ret_sample.values)
             trend = pd.DataFrame(trend, columns=cols, index=idxs)
             return trend
```

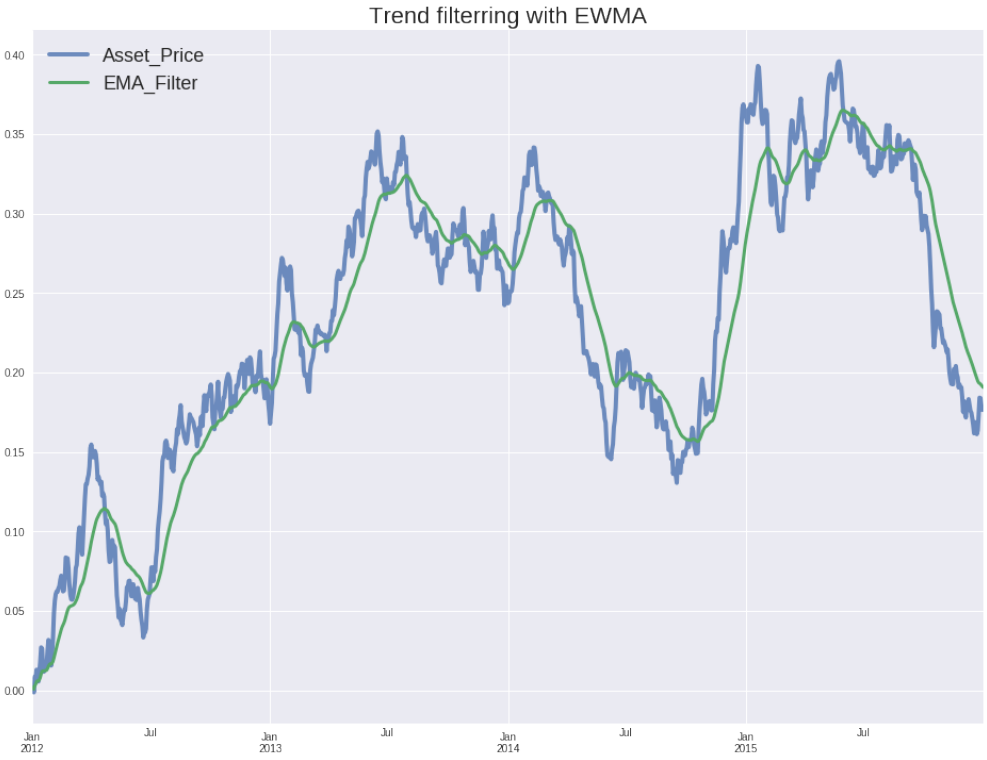
```
def simple_l2_filter(ret_sample, lbd):
    """ Simple test of L2 filter
    """
    n_sample = len(ret_sample)
    L = build_l2_matrix(n_sample, lbd)
    trend = l2_filter(ret_sample, L)
    return trend

def rolling_l2_fiter(ret, lbd, periods=250):
    """ Build rolling l2 filter
    """
    date_list = ret.index
    date_ts = pd.Series(date_list)
    trend = dict()
    L = build_l2_matrix(periods, lbd)
    for k, d in date_ts.ix[periods:].to_dict().items():
        end_date = date_ts.ix[k-1]
        start_date = date_ts.ix[k-periods]
        ret_sample = ret.ix[start_date:end_date]
        tmp_trend = l2_filter(ret_sample, L)
        trend[d] = tmp_trend.ix[end_date]
    return pd.DataFrame(trend).T
```

In [15]:
```
# Test simple one itereation
ret_sample = ret.ix['2010':]
lbd = 200.
test = simple_l2_filter(ret_sample, lbd)
f = figure(figsize=(16, 10))
test['A_2'].ix['2011'].cumsum().plot()
ret_sample['A_2'].ix['2011'].cumsum().plot();
```



### 3.2.4 Example of EMWA filter

```
In [16]: #
         f = figure(figsize=(16, 12))
         ret_sample['A_2'].ix['2012':'2015'].cumsum().plot(
             label='Asset_Price', lw=4, alpha=0.8)
         ret_sample['A_2'].ix['2012':'2015'].cumsum().ewm(20)\
             .mean().plot(label='EMA_Filter', lw=3)
         title('Trend filterring with EWMA', fontsize=22)
         legend(loc=0, fontsize=18);
```

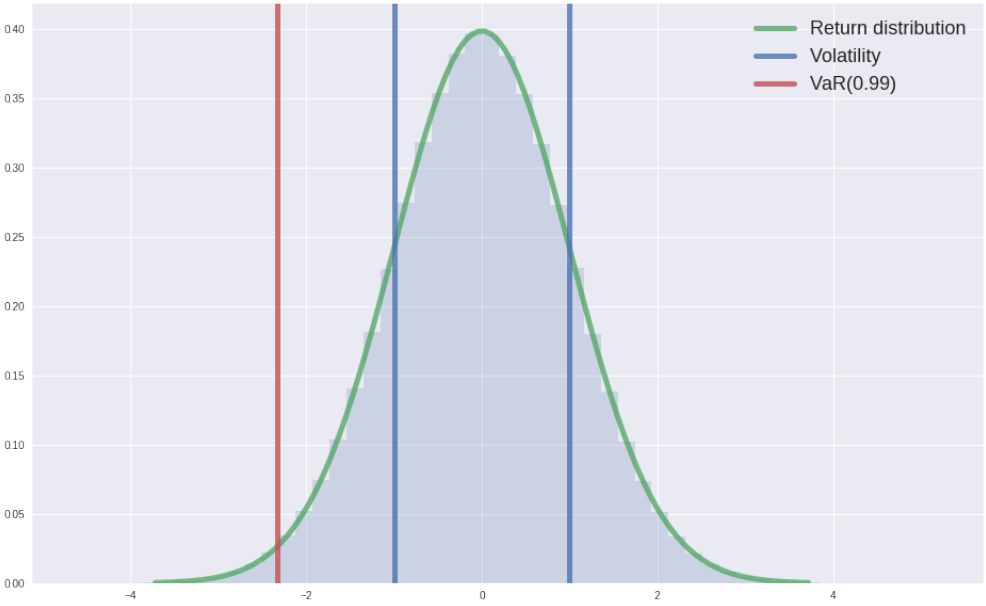



## 3.3 Risk measurement

- A typical example of risk metric is the volatility $\sigma$
- Another useful metric is VaR. It measures the potential loss for a given default probability
- For prob=99% in loss scenario, the potential loss (VaR) = $2.3\sigma$

```
In [17]: #
         from scipy.stats import norm
         x = np.linspace(norm.ppf(0.0001), norm.ppf(0.9999), 100)
         fig = figure(figsize=(16, 10))
         ax = fig.add_subplot(111)
         ax.plot(x, norm.pdf(x), 'g-', lw=5, alpha=0.8, label='Return distribution')
         r = norm.rvs(size=1000000)
         ax.hist(r, normed=True, histtype='stepfilled', alpha=0.2, bins=51)
         ax.axvline(-1, color='b', lw=5, alpha=0.8)
         ax.axvline(1, color='b', lw=5, alpha=0.8, label='Volatility')
```

```
ax.axvline(-2.326, color='r', lw=5, alpha=0.8, label='VaR(0.99)')
ax.legend(loc='best', frameon=False, fontsize=18);
```




### 3.3.1 Volatility measurement

- Historical volatility estimator based on simplified GARCH model:

$$\sigma_t = \sqrt{Ema_T(r_t^2)}$$

  where $r_t = \mathbf{w_t}^\top \mathbf{r_t}$ is the total return of the portfolio.

- Instantaneous volatility estimator:

$$\sigma_t = \sqrt{\mathbf{w_t}^\top \mathbf{\Sigma_t} \mathbf{w_t}}$$

  $\mathbf{w_t}$ is the current position of the portfolio and $\Sigma_t = \mathbb{E}[\mathbf{r_t}^\top \mathbf{r_t}]$ is an estimator of the covariance matrix.

### 3.3.2 VaR parametric

Assuming the PnL has a Gaussian distribution which depends only on the drift $\mu$ and the volatility $\sigma$

$$PnL \sim \mathcal{N}(\mu, \sigma)$$

Using this formula, we can compute the VaR by the following formula

$$VaR(\alpha) = -\mu_{PnL} + \Phi^{-1}(\alpha) \times \sigma$$

Here the VaR is the corresponding quantile for the given threshold $\alpha$ of a probability of the loss distribution. Here we regenerate the result presented in Thierry Roncalli's book on Risk management (page 61-81)

### 3.3.3 VaR with Corner-Fisher expansion

If the distribution has other characteristics such as skewness and kurtosis, we can use the Corner-Fisher expansion in order to approximate the partition function around a Gaussian limit

$$VaR(\alpha) = -\mu + z_\alpha(\gamma_1, \gamma_2) \times \sigma$$

$$z_\alpha(\gamma_1, \gamma_2) = z_\alpha + \frac{1}{6}(z_\alpha^2 - 1)\gamma_1 + \frac{1}{24}(z_\alpha^3 - 3z_\alpha)\gamma_2 - \frac{1}{36}(2z_\alpha^3 - 5z_\alpha)\gamma_1^2 + \dots$$

with $z_\alpha = \Phi^{-1}(\alpha)$ and $\gamma_1, \gamma_2$ excess of skewness and kurtosis (for Gaussian limit, skewness is zero and kurtosis is 3).

### 3.3.4 Implementation of risk measures

In [18]:
```
# Historical and Instantaneous volatility estimators
from sklearn.covariance import ledoit_wolf, oas
from scipy.stats import norm
from scipy.stats import t

def var_gaussian(alpha, mu, sigma):
    """
    Measure the Gaussian VaR
    """
    rv = norm()
    var = -mu + rv.isf(1 - alpha) * sigma
    return var

def var_t(alpha, mu, sigma, nu):
    """
    Measure the Gaussian VaR
    """
    rv = t(nu)
    var = - mu + rv.isf(1 - alpha) * sigma * np.sqrt((nu - 1.) / nu)
    return var

def var_corner_fisher(alpha, mu, sigma, skew, kurt):
    """
    Measure the Corner-Fisger VaR
    """
    rv = norm()
    zalpha = rv.isf(1 - alpha)
    g1 = - skew
    g2 = kurt - 3
    zg1g2 = zalpha + (zalpha**2 - 1) * g1 / 6.\
          + (zalpha**3 - 3 * zalpha) * g2 / 24.\
          - (2 * zalpha**3 - 5 * zalpha) * g1**2
    var = - mu + zg1g2 * sigma
    return var
```

## 3.4 Risk-based portfolio

- We employ here the "Equal risk" allocation allowing a similar fluctuation for all assets.

$$w_i = 1 \,/\, \sigma_i \quad \forall i \in \{1, \ldots, N\} \tag{5}$$

- We fix the annual risk target of the portfolio at the level 10M$
- The portfolio is equilibrium, no asset may cause an extreme loss for the portfolio

```
In [19]: def dict_to_multi_df(df_dict, align='union'):
             """
             Convert a dictionary of dataframe to a multi-index dataframe
             """
             df_list = df_dict.values()
             if align:
                 if align == 'union':
                     indexes = [d.index for d in df_list]
                     union_index = reduce(lambda x, y: x.union(y), indexes)
                 elif align == 'intersection':
                     indexes = [d.index for d in df_list]
                     union_index = reduce(lambda x, y: x.intersection(y), indexes)
                 else:
                     raise ValueError('No align method')
                 df_list = [d.reindex(union_index) for d in df_list]

             result = pd.concat(df_list, keys=df_dict.keys(), axis=1)
             return result
```

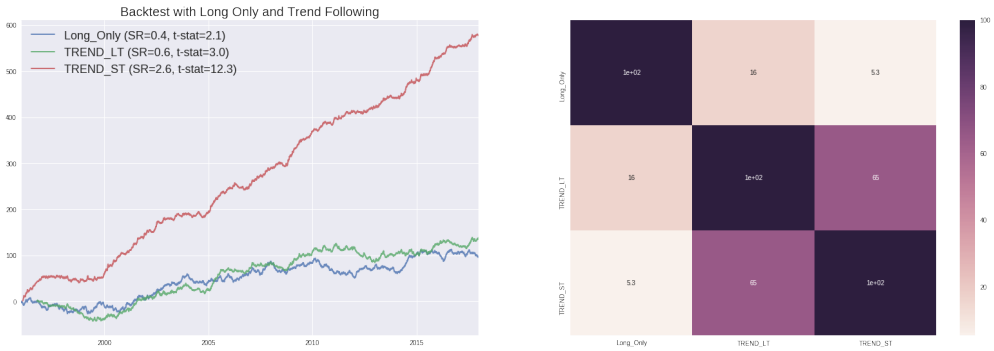
```
In [20]: # Build predictors
         st_scale = 20
         st_preds = ret.ewm(st_scale, min_periods=st_scale).mean().shift(1)
         st_preds = st_preds / st_preds.abs().ewm(250).mean()
         lt_scale = 200
         lt_preds = ret.ewm(lt_scale, min_periods=lt_scale).mean().shift(1)
         lt_preds = lt_preds / lt_preds.abs().ewm(250).mean()
         preds = dict_to_multi_df({'TREND_LT': lt_preds, 'TREND_ST': st_preds})
         risk_target = 10
```

```
In [21]: # Backtest with given risk target
         spnls = preds.shift(1).mul(ret, level=1, axis=1)
         tot_pnls = spnls.groupby(axis=1, level=0).sum()
         tot_pnls = {c: tot_pnls[c] for c in tot_pnls.columns}
         tot_pnls.update({'Long_Only': ret.sum(1)})
         tot_pnls = pd.DataFrame(tot_pnls)
         tot_pnls /= tot_pnls.std()
         tot_pnls *= (risk_target / sqrt(256))
```

### 3.4.1 Backtest

```
In [22]: #
         def plot_pnls(pnls, **kw):
             """ Plot cumulated return with stats
             """
             if isinstance(pnls, pd.Series):
                 pnls = pd.DataFrame({'PnL': pnls})
             pnls.cumsum().plot(**kw)
             srs = np.round(sharpe_ratio(pnls), 1)
             tstats = np.round(t_stats(pnls), 1)
```

```
        legend([c + ' (SR=' + str(srs[c]) + ', t-stat=' + str(tstats[c])
                + ')' for c in pnls.columns], fontsize=18)

f = figure(figsize=(28, 9))
ax = f.add_subplot(121)
plot_pnls(tot_pnls, ax=ax, alpha=0.8, lw=2)
title('Backtest with Long Only and Trend Following', fontsize=20)
ax = f.add_subplot(122)
sns.heatmap(100 * tot_pnls.corr(), annot=True, ax=ax);
```



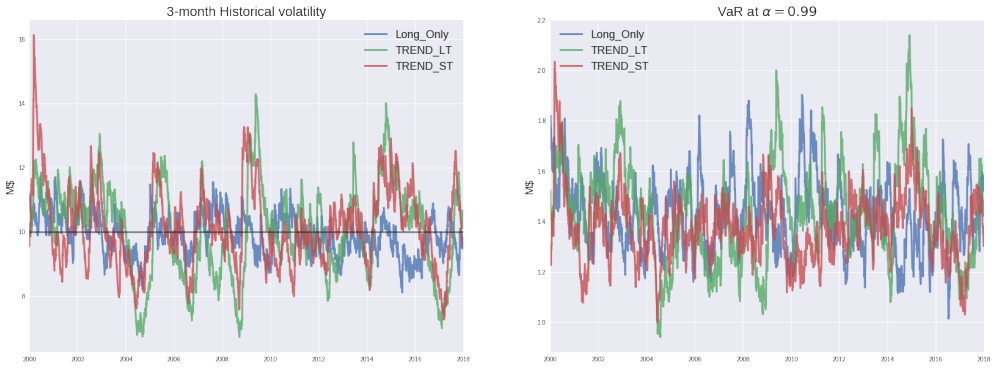

### 3.4.2 Why does trend following works in this example?

- We simulated a price process with:
  - Positive long-term drift
  - Positive short-term auto-correlation in the noise
- Using a long-term trend following will capture the long-term drift
- Using a short-term trend following will capture the short-term positive auto-correlation
- Diversification effect due to trading low-correlated assets

### 3.4.3 Monitoring the portfolio: Risk, VAR

```
In [23]: # Compute historical risk and VaR
         tot_vols = historical_volatility(tot_pnls, 60).ix['2000':]
         tot_vars = dict()
         tot_mus = tot_pnls.rolling(60).mean().ix['2000':]
         alpha = 0.99
         for c in tot_vols.columns:
             tot_vars[c] = dict()
             for d in tot_vols.index:
                 tot_vars[c][d] = var_gaussian(alpha, tot_mus[c][d],
                                               tot_vols[c].ix[d] / sqrt(252))
             tot_vars[c] = pd.Series(tot_vars[c])
         tot_vars = pd.DataFrame(tot_vars)

In [24]: #
         f = figure(figsize=(28, 10))
         ax = f.add_subplot(121)
         tot_vols.plot(ax=ax, lw=3, alpha=0.8)
         title('3-month Historical volatility', fontsize=22)
```

```
ax.set_ylabel('M$', fontsize=18)
ax.axhline(risk_target, color='k', alpha=0.5, lw=3)
legend(fontsize=18)
ax = f.add_subplot(122)
tot_vars.plot(ax=ax, lw=3, alpha=0.8)
title(r'VaR at $\alpha=0.99$', fontsize=22)
ax.set_ylabel('M$', fontsize=18)
legend(fontsize=18);
```




## 3.5 Markowizt portfolio

- Another management style is based on the performance. We consider here the typical example of Markowitz portfolio.
- $\mathbf{w_t}$ the vector of allocation for the set of $N$ asset/strategy, $\bar{}_t$ be the prediction of the performance of the given pair asset/strategy.
- The optimization program can be written as:

$$\begin{aligned} \max_{\mathbf{w}} \quad & \mathbf{w_t}^\top \bar{}_t - \mathbf{fl}\|\mathbf{w_t}\|_{L_1} \\ \text{u.c.} \quad & \mathbf{w_t}^\top \mathbf{\Sigma} \mathbf{w_t} \leq \mathbf{R^2_{target}} \\ & \mathbf{w_t} > \mathbf{0} \end{aligned}$$

  - The performance: $\mathbf{w_t}^\top \bar{}_t$
  - The linear cost: $\gamma\|\mathbf{w_t}\|_{L_1}$
  - The risk of portfolio: $\mathbf{w_t}^\top \mathbf{\Sigma} \mathbf{w_t}$

This is a typical Markowitz problem with an additional term modeling the trading cost.

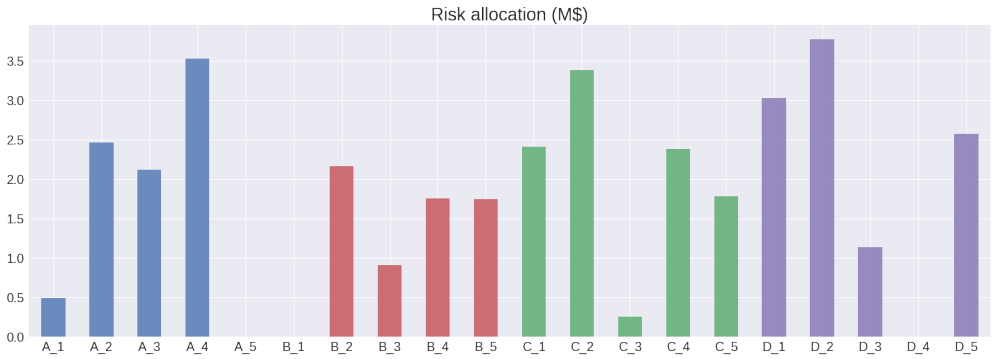
### 3.5.1 Implementation

```
In [25]: def markowitz_porfolio(pnls, lin_cost, os_date, risk_tg= 10):
             """
             Optimization program for Markowitz portfolio
             """
             gain = pnls.ix[:os_date].mean() / pnls.ix[:os_date].std()
             cov = pnls.ix[:os_date].corr()
             col_list = pnls.columns
```

```
    # Declare variables
    col_len = len(col_list)
    cvx_w = cvx.Variable(col_len)
    cvx_gain = cvx.Parameter(col_len)
    cvx_lin_cost = cvx.Parameter(col_len, sign='positive')
    constraints = []

    # Initiate variables
    cvx_lin_cost.value = lin_cost.values
    cvx_gain.value = gain.values

    # Build optimization problem
    util_func = cvx_w.T * cvx_gain - cvx.abs(cvx_w).T * cvx_lin_cost
    obj_func = cvx.Maximize(util_func)

    # Build constraints
    constraints.append(cvx_w > 0.)
    constraints.append(cvx.quad_form(cvx_w, cov.values) <= risk_tg ** 2)

    # Solve optiomization program
    cvx_prob = cvx.Problem(obj_func, constraints)
    cvx_prob.solve(solver=cvx.ECOS)
    optimal_weights = pd.DataFrame(cvx_w.value, index=col_list)[0]
    return optimal_weights / pnls.ix[:os_date].std() / sqrt(252)
```

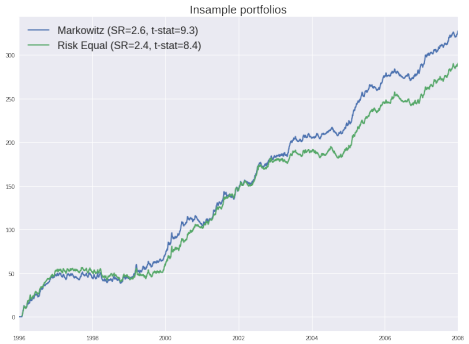
### 3.5.2 Optimal allocation

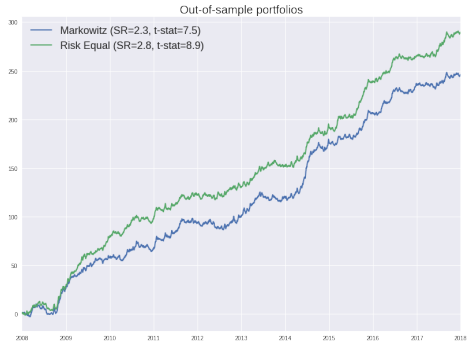
```
In [26]: # Simulation
         mdl = 'TREND_ST'
         risk_target = 10
         os_date = '20080101'
         col_names = spnls[mdl].columns
         unit = pd.Series(ones(len(col_names)), index=col_names)
         lin_cost = (0.2 / sqrt(252)) * unit
         mkw_w = markowitz_porfolio(spnls[mdl], lin_cost, os_date, risk_target)

         # Plot allocation weights
         colors = ['b'] * 5 + ['r'] * 5 + ['g'] * 5 + ['m'] * 5
         f = figure(figsize=(24, 8))
         ax = f.add_subplot(111)
         (mkw_w * spnls[mdl].std() * sqrt(252)).sort_index().plot(kind='bar',
         ax=ax, rot=0, color=colors, fontsize=18, alpha=0.8)
         title('Risk allocation (M$)', fontsize=25);
```




### 3.5.3 Backtest

```
In [27]: #
         f = figure(figsize=(30, 10))
         mkw_pnl = spnls[mdl].mul(mkw_w, axis=1).sum(1)
         cmp_pnls = pd.DataFrame({'Markowitz': mkw_pnl,
                                  'Risk Equal': tot_pnls[mdl]})
         ax = f.add_subplot(121)
         plot_pnls(cmp_pnls.ix[:os_date], ax=ax, lw=2)
         title('Insample portfolios', fontsize=20)
         ax = f.add_subplot(122)
         plot_pnls(cmp_pnls.ix[os_date:], ax=ax, lw=2)
         title('Out-of-sample portfolios', fontsize=20);
```




# 4 Conclusions and Perspectives

## 4.1 Conclusions

- This simple example shows how the systematic asset management works
- Warning: Many important parts are not discussed in this presentation
  - Automatic execution
  - Trading error
  - Trade allocation for different accounts
  - Margin management, tail-risk management

  - ...
- Robot-advisors work based on the similar principle but much more sophisticated
  - Need to clean parameters
  - Need to have better optimization with the real market condition
  - Need to be performant to handle large dimension
  - Need more support for monitoring, stats, backtests

### 4.2 Perspectives

- With a better understanding of a class of traditional trading strategies, a new management style emergers since last few years named *Alternative Risk Premium*[11]:
  - Systematic management with standard strategies such as momentum, carry, risk premium
  - Portfolio construction for reducing trading cost and better risk management
  - This new field is opened to all actors: hedge funds, traditional asset management, advisors, e.t.c
- Fintech plays a central role to improve all aspects of systematic management
  - Can Machine Learning/AI improve the quality of prediction?
  - Stabilize Optimization program with high dimension?
  - Overfitting questions
  - Application to other domain such as *sport betting*
  - Trading Bitcoin as an asset, do not "betting" on bitcoin

## References


[1] RONCALLI, T. (2010), La gestion d'actifs quantitative, *Economica*.

[2] RONCALLI, T. (2018), Risk Management & Financial Regulation, *SSRN*.

[3] WILMOTT, P., Introduces Quantitative Finance, *Wiley*.

[4] BOUCHAUD, J-P. and POTTERS, M., Theory of Financial Risk and Derivative Pricing: From Statistical Physics to Risk Management, *Cambridge University Press* .

[5] BRUDER, B., and GAUSSEL, N. (2011), Risk-Return Analysis of Dynamic Investment Strategies, *SSRN*.

[6] DAO, T-L. (2017), Convexity of Trend following and Variance Arbitrage, *Presentation at Metori Capital Management*.

[7] DAO, T-L., NGUYEN, T.T., DEREMBLE, C., LEMPÉRIÈRE, Y., BOUCHAUD, J-P., and POTTERS, M. (2016), Tail Protection for Long Investors: Trend Convexity at Work, *SSRN*.

[8] BRUDER, B., DAO, T-L., RICHAR, J-C. and RONCALLI, T., (2012), Trend Filtering Methods for Momentum Strategies , *SSRN*.

[9] JUSSELIN, P., LEZMI, H., MALONGO, H., MASSELIN, C., RONCALLI, T., and DAO, T-L. (2017), Understanding the Momentum Risk Premium: An In-Depth Journey Through Trend-Following Strategies, *SSRN*.

[10] DAO, T-L. (2014), Momentum Strategies with L1 Filter, *SSRN*.

[11] HAMDAN, R., PAVLOWSKY, F., RONCALLI, T., and ZHENG, B. (2016), A Primer on Alternative Risk Premia, *SSRN*.