Linear Programming Formulation of Portfolio Management

created by: pgeorge@mesoncapital.com
last edited: 2017-10-10

Overview

This document presents portfolio management, as used by the Gravity simulation engine, as a linear programming problem in standard form. It also discusses implementation details using the [scipy.optimize.linprog](#) Python module.

Data

- [Ideal_portfolio.2010-01-01.csv](#)

CSV headers:

name	type	description
tradingitemid	integer	Internal equity ID.
ind_long	boolean	Indicates long position (proba_long > 0.5).
ind_short	boolean	Indicates short position (proba_short > 0.5).
proba_long	float	Probability long equity (i.e. value will increase).
proba_short	float	Probability short equity (i.e. value will decrease).
marketcap	float	Marketcap (MM).
volume	integer	Weekly trade volume.
beta	float	12 month average volatility relative to S&P 500.
borrowcost	float	Actual annualized cost to borrow.
borrowavailability	integer	Outstanding volume available for borrow.

Mathematical Formulation

Ideal Portfolio

Consider probability vectors $\mathbf{p^L}$ and $\mathbf{p^S}$ generated by a binary classification model, defined such that the elements

$$p_i^L \equiv \text{probability long}$$
$$p_i^S \equiv \text{probability short}$$
$$i \in [0..N-1]$$

For both longs and shorts, we neglect any position with a probability < 0.5.

We apply an exponential weighting function to $\mathbf{p^L}$ and $\mathbf{p^S}$ to generate a set of position weight vectors $\mathbf{\hat{w}^L}$ and $\mathbf{\hat{w}^S}$, respectively. These vectors should be normalized such that

$$\sum_i \hat{w}_i^L = M_L \quad (\hat{w}_i^L > 0)$$
$$\sum_i \hat{w}_i^S = M_S \quad (\hat{w}_i^S < 0)$$
$$M_L \equiv \text{normalized long exposure}$$
$$M_S \equiv \text{normalized short exposure}$$

For example, an unlevered dollar-neutral portfolio would have $M_L = 1.0, M_S = -1.0$.

Objective Function

Given an ideal portfolio, $\hat{P}$

$$\hat{P} = \left[\mathbf{\hat{w}^L}, \mathbf{\hat{w}^S}\right]$$

we seek to minimize the error between it and our optimal portfolio, P

$$P = \left[\mathbf{w^L}, \mathbf{w^S}\right]$$

objective function:

$$\min \quad \sum_{\substack{i \in [0..N-1] \\ j \in (L,S)}} |w_i^j - \hat{w}_i^j|$$

Optimization Constraint Bounds (Optimization Hyperparameters)

This table contains constraints definitions and default values for portfolio optimization. The goal is make these constraint bounds parameterizable inputs to our portfolio management algorithm.

$$w_{max}^L \equiv \text{maximum long position weight} \quad (\text{default} = 0.05)$$
$$w_{min}^S \equiv \text{maximum long position weight} \quad (\text{default} = -0.03)$$
$$M_L \equiv \text{normalized long exposure} \quad (\text{default} = 1.00)$$
$$M_S \equiv \text{normalized short exposure} \quad (\text{default} = -1.00)$$
$$\delta_L^M \equiv \text{long exposure buffer} \quad (\text{default} = 0.1)$$
$$\delta_S^M \equiv \text{short exposure buffer} \quad (\text{default} = 0.1)$$
$$\delta_L^\beta \equiv \text{long portfolio beta buffer} \quad (\text{default} = 0.1)$$
$$\delta_S^\beta \equiv \text{short portfolio beta buffer} \quad (\text{default} = 0.1)$$

Constraints

Here, we itemize the list of constraints.

1. **Bounds on individual weightings**

$$0 < w_i^L \le w_{max}^L$$
$$w_{min}^S \ge w_i^S < 0$$

2. **Bounds on exposure**

$$1 - \delta_L^M \le \frac{1}{M_L} \sum_i w_i^L \le 1 + \delta_L^M$$

$$1 - \delta_S^M \le \frac{1}{M_S} \sum_i w_i^S \le 1 + \delta_S^M$$

3. **Bounds on relative volatility ("beta"). We will eventually revisit this constraint to enforce near-beta-neutrality.**

$$1 - \delta_L^\beta \leq \sum_i \beta_i^L w_i^L \leq 1 + \delta_L^\beta$$

$$-(1 - \delta_S^\beta) \leq \sum_i \beta_i^S w_i^S \leq -(1 + \delta_S^\beta)$$

4. **Constraints on liquidity (to be completed)**

   We should have something here to that say 50% of entire portfolio can be liquidated completely in 2 weeks at 20% of volume cap. Potential hyperparams:

   1. percent of portfolio required to be liquid (default = 50%)
   2. time period of liquidity (default = 2 weeks)
   3. max volume percent (default = 20%)

5. **Constraints on short availability (to be completed)**

   This is a tricky one because you need to define this constraint as the new position (e.g. PositionSizeIdeal - PositionSizeToday = order size <= short avail quantity). I think this is ok to use though since position size today is known.

Scipy Implementation

To be completed.