Available online at www.sciencedirect.com
ScienceDirect
Expert Systems with Applications 35 (2008) 273–278
Expert Systems with Applications
www.elsevier.com/locate/eswa

Dynamic proportion portfolio insurance using genetic programming with principal component analysis ☆

Jiah-Shing Chen *, Chia-Lan Chang, Jia-Li Hou, Yao-Tang Lin

Department of Information Management, National Central University, Jhongli 320, Taiwan, ROC

Abstract

This paper proposes a dynamic proportion portfolio insurance (DPPI) strategy based on the popular constant proportion portfolio insurance (CPPI) strategy. The constant multiplier in CPPI is generally regarded as the risk multiplier. Since the market changes constantly, we think that the risk multiplier should change according to market conditions. This research identifies risk variables relating to market conditions. These risk variables are used to build the equation tree for the risk multiplier by genetic programming. Experimental results show that our DPPI strategy is more profitable than traditional CPPI strategy. In addition, principal component analysis of the risk variables in equation trees indicates that among all the risk variables, risk-free interest rate influences the risk multiplier most.

Keywords: Dynamic proportion portfolio insurance (DPPI); Constant proportion portfolio insurance (CPPI); Genetic programming (GP); Principal component analysis (PCA)

1. Introduction

Constant proportion portfolio insurance (CPPI) is a simple strategy based on the investor's preference of risk to calculate the amount invested in the risky asset (Black & Jones, 1987; Black & Perold, 1992). The rest of capital is invested in the risk-free asset. The purpose of CPPI is to help investors capture the upside potential of the equity market while maintaining a minimum floor of the portfolio.

Because the risk multiplier in the CPPI strategy is predetermined by an investor's own view, inappropriate expectation of the market will result in loss (Hakano, Kopprasch, & Roman, 1989). When the market becomes more volatile, a larger risk multiplier may cause a larger loss. Thus, a new dynamic hedging model needs to be developed (Zhu & Kavee, 1988). Although the concept of CPPI originally requires no volatility estimation for its implementation, choosing the risk multiplier and the floor according to market conditions is necessary and the performance is mostly affected by market trends. An investor should therefore adjust the risk multiplier in CPPI to adapt to market conditions.

As a result, this paper proposes a new portfolio insurance strategy, named dynamic proportion portfolio insurance (DPPI), based on the risk variables which are used to build the equation tree for the risk multiplier by genetic programming. The performance of DPPI is compared to the traditional CPPI. In addition, principal component analysis is performed to determine the most important risk variables for the risk multiplier.

The rest of this paper is organized as follows. Section 2 reviews the research on constant proportion portfolio insurance (CPPI), genetic programming and principal component analysis. Section 3 describes our proposed dynamic proportion portfolio insurance (DPPI). The experimental process and experimental results are presented in Section 4. The concluding remarks and future directions are stated in Section 5.

☆ This work was supported by a grant (NSC95-2416-H-008-030) from the National Science Council of Taiwan.
* Corresponding author. Fax: +886 3 4254604.
E-mail addresses: jschen@ncu.edu.tw (J.-S. Chen), lans@mgt.ncu.edu.tw (C.-L. Chang), alexhou@mgt.ncu.edu.tw (J.-L. Hou), s0443003@cc.ncu.edu.tw (Y.-T. Lin).

0957-4174/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2007.06.030

2. Background

2.1. Constant proportion portfolio insurance

Black and Jones (1987) introduced the constant proportion portfolio insurance (CPPI) strategy. The portfolio contains an active asset and a reserved risk-free asset. The active asset has a higher expected return than the reserved asset. For example, the active asset can be a stock while the reserved asset might be a T-bill. The CPPI strategy can be expressed by the following formula:

$$E = M \times (A - F),$$

where exposure E is the dollar amount invested in the active asset, asset A is the value of the portfolio, and floor F is the lowest acceptable value of the portfolio. The difference between asset and floor computes the cushion as the excess of the portfolio value over the floor. In addition, the risk multiplier M represents the ratio of the initial exposure to the initial cushion. When the active asset changes in value, the portfolio value will change accordingly and the cushion is also different from its initial value. The variable cushion is then multiplied by the constant risk multiplier, which is predetermined by the investor, to get the new exposure of the active asset.

This simple formula is easily understood by investors and helps them design their own strategies to meet their preferred level of risk. It is also straightforward to implement. However, determining the best risk multiplier to obtain a better investment result is a difficult task.

Genetic algorithms (GA) can be used to optimize the risk multiplier in CPPI in stead of using the investor's preference. However, the GA-optimized risk multiplier is still a constant which cannot adapt to the changing market conditions.

2.2. Genetic programming

Genetic Programming (Banzhaf, Nordin, Keller, & Francone, 1998; Koza, 1990, 1992, 1994a, 1994b) is a recent development which extends classical genetic algorithms (Bäck, Fogel, & Michalewicz, 2000a, 2000b; Goldberg, 1989; Holland, 1992; Mitchell, 1996) to process nonlinear structures. This optimization technique is based on the principles of natural evolution and is consisted of several genetic operators: selection, crossover, and mutation.

The major difference between genetic programming and genetic algorithms is the representation of the solution candidates. A hierarchical tree structure (as depicted in Fig. 1) represents a solution candidate in genetic programming while a string of characters with a fixed length represents a solution candidate in genetic algorithms.

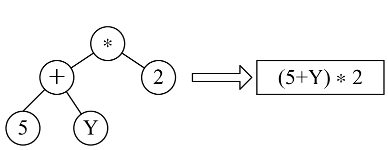

Fig. 1. A tree in genetic programming represents a formula.

The genetic programming framework consists of the flowing elements: node definition, initialization, fitness evaluation, selection, crossover, mutation, and termination condition.

Node definition: The nodes in the tree structure of genetic programming can be classified into two types. One of them is the terminal set which is consisted of constants or variables. The other one is the function set which is consisted of standard arithmetic operations, standard programming operations, standard mathematical functions, logical functions, or domain-specific functions. The elements of the terminal set and the function set are used to construct well-formed expression trees, which represent solutions to the problem, according to certain rules.

Initialization: Genetic programming starts with an initial population of expression trees which are randomly generated.

Fitness evaluation: Fitness evaluates how well a tree performs in the problem environment. Fitness values are used by the selection method to select trees for reproduction.

Selection: The selection method determines how trees are selected from the population to be parents for crossover. Better parents are usually selected with the hope that they have a better chance of producing better offsprings. The roulette wheel selection is a popular selection method.

Crossover: In GP, subtree crossover is the most common crossover operator. Subtree crossover works by replacing a subtree in one parent with a subtree from the other parent to produce the offspring. Fig. 2 presents an example of subtree crossover.

Mutation: In GP, mutation is usually achieved by replacing a subtree with a randomly generated subtree or by exchanging two randomly selected subtrees. An example of subtree replacement mutation is presented in Fig. 3.

Termination condition: Common termination conditions include fixed generations, fitness target, fitness convergence, and diversity convergence.

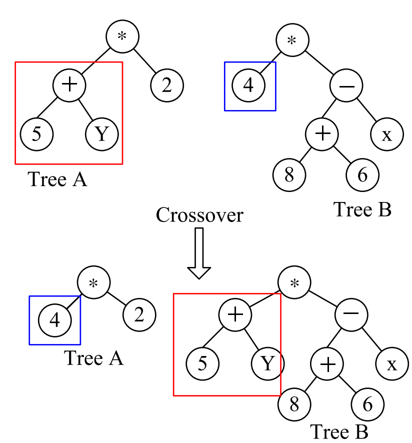

Fig. 2. An example of subtree crossover.