PSL: Portfolio Simulation Language
White Paper

Alexander Besidski
ab2012@columbia.edu

Wei-Chen Lee
wl2135@columbia.edu

Xin Li
xl74@columbia.edu

Jian Huang
jh2353@columbia.edu

September 27, 2004

1 Introduction

Over the last few decades, the average person's interest in the stock market has grown exponentially. What was once a toy of the rich has now turned into the vehicle of choice for growing wealth. This demand coupled with advances in trading technology has opened up the markets so that nowadays nearly anybody can own stocks. While there exist various tools designed for creating complex financial models based on market data, these tools are all geared towards sophisticated business users. Portfolio simulation language is designed specifically for the purpose of simulating and modeling behaviors of financial assets in a way accessible to inexperienced users having knowledge of only a few fundamental concepts. In addition to being easily accessible the language is also designed to be flexible for the more advanced investors. As a simple programming language, it is particularly suitable as a flexible and convenient teaching tool for demonstrating and practicing basic financial market operations for students majoring in finance.

2 Fundamentals

Two concepts will lie at the heart of our language — portfolio and asset. Portfolios are essentially collections of assets with a few additional attributes. PSL will define constructs to manipulate portfolio content and define functions to operate on them. Two examples of assets are stocks and bonds. At the very minimum all assets share two common characteristics — rate of return and variance. Having these two attributes at the core of every asset will allow us to create portfolios comprised of different types of assets and to run meaningful analysis on their content, which is the goal of this project. By making financial instruments into first class citizens of our language we will obtain the ability to define portfolios with ease and run various types of operations with a few lines of code.

3 Features

3.1 Simplicity

Since we would like to create a language that is easy enough for a user who is inexperienced in both programming and finance the language will have only a few basic types in addition to assets and portfolios primarily added in order to manipulate assets and create custom functions.

3.2 Hierarchical

Being an intrinsically object-oriented programming language, PSL adopts a hierarchy of data types. Each of these data is customized by different properties (data members) and behaviors (member functions), while all of them sharing certain set of common characteristics. Basically, three hierarchical levels of abstraction exists:

a. primitive data types
   These basically include numbers, logicals, strings, and constants, tosupport basic arithmetic and logical operations, which is indispensable for implementing any full-fledged programming languages.

b. financial instruments (assets)
   These include [stock], [bond] and [cash]. Here [cash] is modeled as a riskless financial instrument having a fixed rate of return. [bond] is similar to [cash] except that rather than simply compounded, it has a certain maturity date and a fixed coupon yield. At maturation [bond] is automatically converted to [cash]. [stock] has a property named volatility to measure its risk, as well as an expected rate of return. The stochastic behavior of [stock] can be modeled by lognormal distribution which is implemented in the member functions of the [stock] data type.

c. financial derivatives
   These are basically more complicated financial instruments derived from the basic financial instruments in (b), including [option] and [future], which must have an underlying asset (financial instrument) in order to be fairly priced. Due to the intrinsic stochastic nature of the underlying assets (stocks), simulation and modeling operations are also supported for financial derivatives.

d. portfolio
   This is the highest level data type in PSL, and lies at the core of the language. Basically [portfolio] is a composite data type, consisting of arbitrary combinations of (b) and (c) (satisfying certain constraints). A [portfolio] data type supports many operations include risk/return optimization and VaR analysis, as well as Monte Carlo simulation.

In addition, the array type, which consisting of fixed number of homogeneous data types, is supported for all the basic data types in (a)-(d).

3.3 Flexibility

PSL aims to cater to make the language accessible and useful for users with various levels of expertise in programming experience and financial knowledge. It is designed such that users with the minimum knowledge of computer programming would quickly master the basic language elements and be able to write practical programs in PSL. Although internally designed as totally objected-oriented, PSL provides users with a more traditional procedure based programming interface. One could simply write a script-like program, using all the normal features supported by most other languages such as loops, conditional branches, etc, without any need to know anything about objected-oriented programming concepts. Users will be able to construct complicated portfolios and perform highly professional analysis, using the built-in financial instrument types such as [stock], [bond], [future], etc. For most common practices in finance these should be enough. For more advanced users, e.g., traders and financial analysts who may want to manipulate more complicated financial instruments in the portfolio, PSL provides them with the options to build up their own financial derivatives by expanding the basic financial instrument types. This

would require the users to know more about the advanced features of PSL, particularly the objected oriented concepts, since the user have to define their new object types and specify their properties (data members) and behaviors (member functions). However, this is by no way a requisite to master the basic programming in PSL.

3.4 Finance-oriented

PSL is mainly designed to perform financial analysis and portfolio management/simulation. So manipulation of finance instruments and their compositions lies at the heart of the language. PSL provides many built-in financial analysis tool, imbedded in the basic financial instrument types as built-in functions. For example, PSL provides tools for portfolio risk/return optimizations, VaR analysis, historical data extraction, stock movement modeling, option/future pricing, and Monte Carlo simulation of stocks/portfolio paths, to name a few. All these powerful operations provides simple and user-friendly interfaces to the users and all the mathematical details are hidden. Users with little or no mathematical/financial background would still be able to write complicated financial analysis software using our PSL language.

4 Summary

The PSL language provides a simple yet powerful programming tool for the simulation and analysis of financial instruments and portfolios. Everything is internally designed as objected-oriented, while PSL provides users with a procedure-based interface that is sufficient to achieve most common tasks in financial analysis. It is both easy to learn and use, yet flexible enough to achieve very powerful goals. PSL is suitable for a diverse group of users such as finance majoring students/instructors, traders, mutual fund/hedge fund managers, etc.