Using Portfolio Theory for Better and More Consistent Quality

Ken Koster
Agitar Software Laboratories
450 National Avenue, Suite A
Mountain View, California 94043
kenk@agitar.com

ABSTRACT

The effectiveness of software quality techniques varies. Many uncertain or unpredictable factors influence effectiveness, including human factors, the types of defects in the program, and luck. Compared to using a single quality technique, a diversified portfolio of techniques will typically be more effective and less variable. This work postulates a simple model, adapted from financial Modern Portfolio Theory, for the variability and effectiveness of techniques, singly and in portfolios. Proofs and simulations analyze the model to evaluate factors influencing the success of diversification; the model is checked against data sets from previous work.

Categories and Subject Descriptors

D.2.4 [**Software Engineering**]: Software/Program Verification; D.2.9 [**Software Engineering**]: Management

General Terms

Reliability, Economics, Theory

Keywords

Diversification, effectiveness, economic models, portfolio, software quality, testing, variability

1. INTRODUCTION

The optimal software quality strategy aims not only for a low residual defect count, but to achieve high quality with *consistency and reliability*. Technique effectiveness varies; the defects addressed by a technique change from program to program and from practitioner to practitioner. Allocating all quality resources to a single technique – even a highly effective one – may expose program quality to high risk; if that technique – for any number of reasons – happens to perform badly or has inherent limitations that preclude it from addressing some of a program's defects, the overall quality of the project will suffer. While effectiveness should be the main focus of a quality strategy (variable good quality is much better than invariably bad quality), variability of quality should not be disregarded.

Previous work (reviewed below) has established that combining defect detection techniques generally improves effectiveness. We shall investigate how the interplay between different quality techniques affects both variability and effectiveness.

Specifically, this work seeks to expand on Littlewood et al.'s findings that portfolio effectiveness increases with increasing diversity between techniques [9]. That work used the following two simplifying assumptions: (1) that techniques would perform differently on different programs, but would perform with similar effectiveness overall; (2) that techniques would be allocated equal resources. The model presented herein, adapted from financial Modern Portfolio Theory, is structured to enable reasoning about different technique effectiveness levels, different technique resource allocations, and different technique variablity of effectiveness.

This paper does not evaluate the merits of any particular technique. It assumes that software professionals will have beliefs (possibly, but not necessarily based on empirical evidence) about how techniques at their disposal perform and how the effectiveness of those techniques varies. Given those beliefs, this paper seeks to answer the question: "How can techniques be combined to increase effectiveness and improve the consistency and predictability of quality?"

1.1 Previous Work

Most research on software quality techniques has focused on the effectiveness of a single technique; little has considered using techniques in concert [9]. The consensus among this handful of findings is that certain techniques are better at finding certain types of defects, and that combining different, complementary techniques improves effectiveness. Myers combined the results of individuals using structural and functional testing and found significant increases in average fault detection among combinations [11]. Selby also found that the average effectiveness of pairwise combinations of functional testing, structural testing, and code review topped the effectiveness of using those techniques alone [15]. Wood et al. ran experiments on the effects of combining code reading, functional testing, and structural testing, and found that the combination of those techniques was much more effective than using them individually [18]. Porter and Votta found that running specification inspections with dif-

ISSTA'07, July 9–12, 2007, London, England, United Kingdom.

ferent techniques in parallel greatly increased the fault detection rate [12].

One experiment run by Laitenberger dissents [7]. It studied the effect of performing structural testing after code inspection and found that structural testing added little value after a code inspection. Tellingly, the only work reviewed here to have found that combining techniques was ineffective is also the only work to have applied the techniques sequentially rather than running them in parallel.

Some theoretical work has also considered technique combination. Holzmann theorized that combining formal verification techniques with the "standard approach" to testing software would reduce levels of catastrophic residual defects [3]. Of particular relevance to this work, Littlewood et al. provided a convincing probabilistic model of technique effectiveness and diversity, for which they proved that the effectiveness of technique combination improves with increasing diversity between techniques [9].

However, studies of quality techniques – whether considering them individually or in combination – have seldom considered the variability of their effectiveness. Some have considered variability for the sake of statistical validation of experiments [7]. Littlewood et al. considered variance, but only as a measure of diversity and of its consequent impact on effectiveness, not as an important outcome in its own right.

Exceptionally, some have commented on technique variability and its undesirability [11, 18, 8]. Myers found that there was "tremendous variability" in the performance of subjects, while Wood et al. found that "techniques do not perform uniformly and the program (and hence the faults) have an impact on the effectiveness of the technique." Lauterbach and Randall found that the tester/code sample was an especially potent source of variability. However, how to mitigate this variability has not been explored.

1.2 Modern Portfolio Theory

A rich body of work investigates the analogous problem of optimizing for performance and variability within resource constraints for financial investments. Financial Modern Portfolio Theory (MPT) considers how best to allocate resources to securities. Like software quality techniques, securities have return (effectiveness) and risk (variability) components. The return of a security gauges how its value will increase (or decrease) over time. Its risk is the volatility of its returns (typically taken to be the square root of the variance of its returns). Given probability beliefs about the expected returns and risks of a set of securities, MPT can be used to determine the set of asset allocations (portfolios) that efficiently maximize return while minimizing risk.

Laying the foundation of MPT, Markowitz demonstrated that although the expected return of a diversified portfolio of securities was the linear combination of their respective expected returns, the expected risk of that diversified portfolio would be less than the linear combination of their respective expected risks [10]. This holds as long as the returns of the component securities are not perfectly positively correlated (i.e., two or more of the securities have a correlation coefficient (ρ) less than one).

To summarize, let R_i be the return on i^{th} security. Let X_i be the percentage of assets allocated to the i^{th} security, such that all $X_i \geq 0$ and $\sum X_i = 1$. Let σ_{ij} be the covariance between security i and security j. Then the expected return (R) of the whole portfolio of N securities is

$$R = \sum R_i X_i \tag{1}$$

and the variance $Var(R)$ is

$$Var(R) = \sum_{i=1}^{N} X_i^2 Var(R_i) + 2 \sum_{i=1}^{N} \sum_{j>1}^{N} X_i X_j \sigma_{ij} \tag{2}$$

Let σ_i be the standard deviation of the i^{th} security, and ρ_{ij} be the correlation coefficient between security i and security j. Then the two security case, for example, reduces to:

$$R = R_i X_i + R_j X_j \tag{3}$$

$$Var(R) = X_i^2 Var(R_i) + X_j^2 Var(R_j) + 2 X_i X_j \sigma_i \sigma_j \rho_{ij} \tag{4}$$

Note that since $Var(R_i) = \sigma_i^2$, we obtain the following equations when $\rho_{ij} = 1$:

$$Var(R) = X_i^2 \sigma_i^2 + X_j^2 \sigma_j^2 + 2 X_i X_j \sigma_i \sigma_j \tag{5}$$

$$Var(R) = (X_i \sigma_i + X_j \sigma_j)^2 \tag{6}$$

$$\sigma = X_i \sigma_i + X_j \sigma_j \tag{7}$$

In other words, the portfolio risk is a linear combination of the component securities' risks only in the case where $\rho_{ij} = 1$. In all other cases, it is less than the linear combination of risk.

The distinction between portfolio risk and linear combination of risk is a subtle and important one. The linear combination of risk between security i and security j could be sampled by running n experiments, $X_i n$ of which invest only in security i, and $X_j n$ of which invest only in security j. To sample the portfolio risk, all n experiments would invest fraction X_i of resources in security i and fraction X_j of resources in security j.

"Hedging" effects reduce portfolio risk compared to linear combination of risk. For example, if in any of the n experiments, the return of j was greater than expected while the return of i was less than expected, their "unexpectedness" would, in a matter of speaking, offset each other, and bring the portfolio return closer to the expected return, reducing volatility.

Applying MPT, one can determine the set of resource allocations that maximize return for a given level of risk or that minimize risk for a given level of return. This set of allocations is named the "efficient frontier." From a purely objective point of view, all allocations on the efficient frontier are equally "good." Investors would select an allocation from the efficient frontier on the basis of how risk-averse or risk-tolerant they are.

2. MODELING EFFECTIVENESS

2.1 Effectiveness of a Single Technique

Before modeling the effectiveness and variability of quality portfolios, we shall first model the effectiveness of a single technique. Let i correspond to a single software development technique. Let $R_i(t)$ be the fraction of system defects addressed by allocating t resources to technique i. If t resources allocated to technique i addresses all system defects,

then $R_i(t) = 1$; if it addresses none, then $R_i(t) = 0$. We shall postulate and analyze models of how $R_i(t)$ changes at different resource levels.

Unlike financial securities, the expected return of technique i, $R_i(t)$, is not a homogeneous function. Where in the financial case, for $k > 0$, $R_i(kt) = kR_i(t)$, in the technique case, $R_i(kt) \neq kR_i(t)$. In other words, there is no theoretical upper limit to the return of a security (e.g., a security can gain more than 100% of its value in a given time period), but the number of defects found or prevented by a technique cannot be greater than the potential number of defects in a system. In fact, for quality techniques, we would expect $R_i(kt) \leq kR_i(t)$, given that $R_i(kt)$ can never be greater than one.

For example, say a project spends ten hours performing code reviews, during which 50% of the system defects are addressed. An additional ten hours of code review would address at most the remaining 50% of the system defects, though likely would address less than that.

We shall model $R_i(t)$ as a diminishing return function, such that each additional increase of t results in smaller and smaller increases in $R_i(t)$. (In this case, the inequality from above, $R_i(kt) \leq kR_i(t)$, holds for all positive t and k.) As Littlewood et al. put it, there is "a lack of independence in the effects of successive applications of the same fault detection procedure, and this implies a law of diminishing returns."

Definition 1. The Basic Diminishing Model

$$\begin{array}{lll} R_i(t+\epsilon) - R_i(t) > & 0 & \{\epsilon > 0\} \\ R_i(t+\epsilon) - R_i(t) < & R_i(t) - R_i(t-\epsilon) & \{\epsilon > 0, t > \epsilon\} \end{array}$$

Let us return to our code review example and use the Basic Diminishing Model to estimate the productivity of performing an eleventh hour of code review. The first ten hours had addressed 50% of the system defects. We will say that the last of those ten hours addressed 4% out of the 50%.

$$\begin{aligned} R_i(9) &= 0.46 \\ R_i(10) &= 0.5 \end{aligned}$$

For $\epsilon = 1$

$$\begin{array}{lll} 0 < & R_i(10+\epsilon) - R_i(10) < & R_i(10) - R_i(10-\epsilon) \\ 0 < & R_i(11) - R_i(10) < & R_i(10) - R_i(9) \\ 0 < & R_i(11) - R_i(10) < & .4 \end{array}$$

According to the Basic Diminishing Model, we would expect the eleventh hour of code review to address between 0% and 4% of the system defects.

We shall also model the effectiveness of quality techniques with a more specific model. In the proposed *Exponentially Diminishing Model*, linear increases in resource allocation result in exponentially diminishing increases in effectiveness. This model is largely equivalent to the "Exponential Cost Function" used by Wagner [17].

Definition 2. The Exponentially Diminishing Model

$$R_i(t) = 1 - \lambda_i^t \ \{0 < \lambda_i < 1, t \geq 0\}$$

In this model λ_i corresponds to the rate at which defects are addressed by technique i. The lower the λ_i, the higher the rate at which defects are addressed by the technique. We shall call it the *Defect Address Rate (DAR)*.

In contrast to the Littlewood model, this model gives the flexibility to vary resource allocation and technique effectiveness. While the Littlewood model richly represents the probabilistic nature of fault location, and elegantly captures the divergence of technique performance on particular defects, it cannot readily vary overall technique effectiveness or vary resource allocation between techniques. The Exponentially Diminishing Model has the opposite properties.

When applied to our code review example, ten hours spent addressing half of the system defects corresponds to a 0.933 DAR ($1 - 0.933^{10} \approx 0.5$). According to the Exponentially Diminishing Model, eleven hours of code review will address 53% percent of the system defects ($1 - 0.933^{11} \approx 0.53$).

2.2 Effectiveness of a Portfolio

We shall now consider the effectiveness of a diversified portfolio of techniques, R, and how it relates to the R_i of its component techniques.

Let X_i be the fraction of resources allocated to technique i. X_i shall be expressed as a real number between 0 and 1. The expected effectiveness of a portfolio $R(t)$ is

$$R(t) = \sum R_i(X_i t) - \omega$$

The ω term accounts for *overlap*, the defects that would have been found by several techniques. A single addressed defect should contribute to the overall portfolio $R(t)$ only once; a second or third technique addressing an already addressed defect does not improve effectiveness. Let a single defect be d percent of the total number of defects in a system. If that defect would have been addressed by z techniques operating independently, comprising dz of the $\sum R(X_i t)$ term, then the overlap term should be increased by $d(z-1)$ to eliminate double counting.

For example, let us say technique i is code review and technique j is random testing, and our quality portfolio consists of 30% code review and 70% random testing ($X_i = 0.3$ and $X_j = 0.7$). For a total portfolio allocation of 10 hours, 3 hours will be spent on code review and 7 hours will be spent on random testing, so $R(10) = R_i(3) + R_j(7) - \omega$. If 3 hours of code review independently addresses 25% of the system's defects, 7 hours of random testing independently addresses 40% of the system's defects, and 5% are addressed by both random testing and code review, then the portfolio effectiveness is $R(10) = 0.25 + 0.40 - 0.05 = 0.60$.

THEOREM 1. *A quality portfolio is at least as effective as the linear combination of allocating all resources to its component techniques if the techniques behave according to the Basic Diminishing Model and have no overlap. Formally, if the effectiveness of all of the techniques i in a portfolio perform according to the Basic Diminishing Model, and $\omega_{ij} = 0$ for all $i \neq j$, and $X_i > 0$ for all i, then $R(t) \geq \sum X_i R_i(t)$.*

PROOF. We shall prove the case where $\frac{1}{X_i} \in N$ for all i, with the understanding that the reasoning is similar for the more general case. If we split total effort t into n chunks of ϵ effort, such that $t = n\epsilon$ and $nX_i = 1$, we obtain

$$R_i(t) = \sum_{m=1}^{n} \left[R_i\left(m\epsilon\right) - R_i\left(\left[m-1\right]\epsilon\right) \right]$$

From the definition of the Basic Diminishing Model, we have

$$\begin{aligned} R_i(\epsilon) > & \quad R_i(m\epsilon) - R_i([m-1]\epsilon) \\ R_i(t) < & \quad \sum_{m=1}^{n} R_i(\epsilon) \\ < & \quad nR_i(\epsilon) \end{aligned}$$

Since $n = \frac{1}{X_i}$ and $\epsilon = X_i t$,

$$\begin{aligned} X_i R_i(t) < & \quad R_i(X_i t) \\ \sum_i X_i R_i(t) < & \quad R(t) \end{aligned}$$

□

This result shows that when techniques have no overlap, it is better to combine techniques in a portfolio than to probabilistically allocate all resources to a single technique (the linear combination case). Returning to our example portfolio of code review and random testing, this result means that (assuming the techniques conform to the Basic Diminishing Model and they have no overlap) it is better to allocate 3 hours to code review and 7 hours to random testing than to allocate 10 hours to code review with a 30% probability and to allocate 10 hours to random testing with a 70% probability.

At this point, it will be useful to define a new term: *coincidence*. Coincidence is the rate of overlap between the returns of two techniques. Two techniques that address many of the same defects would have a high coincidence; techniques addressing different defects would have a low coincidence. For instance, structural testing and code review would have a lower coincidence than two types of structural testing.[1]

Coincidence is closely related to correlation. Two techniques whose correlation coefficient approaches -1 would be expected to have a negligible, near-zero coincidence.

Since all $R_i(X_i t)$ are real numbers between 0 and 1, for randomly coincident techniques i and j, the overlap ω_{ij} is equal to $R_i(X_i t)R_j(X_j t)$. Thus, for the two-technique case and random coincidence, we have

$$R(t) = R_i(X_i t) + R_j(X_j t) - R_i(X_i t)R_j(X_j t)$$

For example, let us assume that code review (technique i) and random testing (technique j) are randomly coincident. For $R_i(3) = 0.25$ and $R_j(7) = 0.40$, then $\omega = 0.25 \times 0.40 = 0.10$. The portfolio combining three hours of code review with seven hours of random testing would have an effectiveness of $R(10) = 0.25 + 0.40 - 0.10 = 0.55$.

THEOREM 2. *Diversification between two techniques with different DARs that conform to the Exponentially Diminishing Model yields effectiveness superior to the linear combination of those techniques if the techniques have no more than random coincidence. Formally, if R_i and R_j perform according to the Exponentially Diminishing Model such that $0 < \lambda_{i,j} < 1$, and $\lambda_i \neq \lambda_j$, and t resources are allocated between i and j such that $t > 0$, $X_i > 0$, $X_j > 0$, and $X_i + X_j = 1$, then the expected effectiveness of the portfolio of i and j, $R(t)$, is greater than the expected effectiveness of the linear combination of individual techniques i and j, i.e. $R(t) > X_i R_i(t) + X_j R_j(t)$.*

PROOF. Let $L(t)$ correspond to the expected linear combination of the returns of techniques i and j. If we show that $R(t) - L(t) > 0$, then we have proved the result.

$$\begin{aligned} R(t) = & \quad (1 - \lambda_i^{X_i t}) + (1 - \lambda_j^{X_j t}) - (1 - \lambda_i^{X_i t})(1 - \lambda_j^{X_j t}) \\ = & \quad 2 - \lambda_i^{X_i t} - \lambda_j^{X_j t} - 1 + \lambda_i^{X_i t} + \lambda_j^{X_j t} - \lambda_i^{X_i t}\lambda_j^{X_j t} \\ = & \quad 1 - \lambda_i^{X_i t}\lambda_j^{X_j t} \\ L(t) = & \quad X_i(1 - \lambda_i^t) + X_j(1 - \lambda_j^t) \\ = & \quad X_i - X_i\lambda_i^t + X_j - X_j\lambda_j^t \\ = & \quad 1 - X_i\lambda_i^t - X_j\lambda_j^t \end{aligned}$$

$$\begin{aligned} R(t) - L(t) = & \quad (1 - \lambda_i^{X_i t}\lambda_j^{X_j t}) - (1 - X_i\lambda_i^t - X_j\lambda_j^t) \\ = & \quad X_i\lambda_i^t + X_j\lambda_j^t - \lambda_i^{X_i t}\lambda_j^{X_j t} \end{aligned}$$

If $\lambda_i > \lambda_j$, Let $\lambda_i = k\lambda_j$. Then

$$\begin{aligned} R(t) - L(t) = & \quad X_i(k\lambda_j)^t + X_j\lambda_j^t - (k\lambda_j)^{X_i t}\lambda_j^{X_j t} \\ = & \quad \lambda_j^t(X_i k^t + X_j) - k^{X_i t}\lambda_j^{(X_i t + X_j t)} \\ = & \quad \lambda_j^t(X_i k^t + X_j) - k^{X_i t}\lambda_j^t \\ = & \quad \lambda_j^t(X_i k^t + X_j - k^{X_i t}) \end{aligned} \tag{8}$$

Expression 8 corresponds to the difference between the expected diversified return and the expected linear combination of returns. For $t = 0$, expression 8 evaluates to zero (unsurprisingly, since $R(0) = L(0) = 0$). For $t > 0$, if expression 8 is monotonically increasing (i.e., its first derivative is positive), then $R(t) > L(t)$ for all $t > 0$.

$$\begin{aligned} D_t[\lambda_j^t(X_i k^t + X_j - k^{X_i t})] = & \quad X_i k^t \ln k - k^{X_i t} \ln k^{X_i} \\ = & \quad X_i k^t \ln k - X_i k^{X_i t} \ln k \\ = & \quad X_i \ln k(k^t - k^{X_i t}) \end{aligned} \tag{9}$$

Since $k > 1$ and $0 > X_i > 1$, expression 9 is greater than zero for all $t > 0$. The case where $\lambda_i < \lambda_j$ proceeds similarly. □

We shall illustrate Theorem 2 with an example portfolio of code review (technique i) and random testing (technique j). As in previous examples, this example portfolio allocates 30% of resources to code review ($X_i = 0.3$) and the rest to random testing ($X_j = 0.7$). Also like before, for a total portfolio allocation of 10 hours, we will say that code review would independently address 25% of system defects over three hours ($R_i(3) = 0.25$) and random testing would independently address 40% of system defects over 7 hours ($R_j(7) = 0.40$). Since random coincidence between techniques is an assumption of Theorem 2, we also have $\omega = 0.4 \times 0.25 = 0.1$. The portfolio effectiveness is thus $R(10) = R_i(3) + R_j(7) - \omega = 0.25 + 0.4 - 0.1 = 0.55$.

To calculate the linear combination of code review and random testing, we require both techniques' DARs. In accordance with the assumptions of Theorem 2, we assume that both techniques behave according to the Exponentially Diminishing Model. Since $1 - 0.9086^3 \approx 0.25$, $\lambda_i = 0.9086$ and since $1 - 0.9296^7 \approx 0.4$, $\lambda_j = 0.9296$. The effectiveness

[1]Foreman and Zweben report an interesting real-world example of coincidence [2]. Out of thirty faults in a particular version of TeX, thirteen were detected by all-uses coverage testing. Branch coverage testing detected eleven of those thirteen faults, but none of the other seventeen.

of the linear combination of code review and random testing is

$$\begin{aligned} L(10) = & \quad X_i(R_i(10)) + X_j(R_j(10)) \\ = & \quad 0.3[1-\lambda_i^{10}] + 0.7[1-\lambda_j^{10}] \\ = & \quad 0.3(1-0.9086^{10}) + 0.7(1-0.9296^{10}) \\ = & \quad 0.547 \end{aligned}$$

Thus, the linear combination of code review and random testing (choosing to allocate all resources to code review 30% of the time and to random testing with a 70% of the time) performs less well (0.547) than the portfolio of those two techniques (0.55). (The portfolio effectiveness is 0.5500 when recalculated with the approximate DARs determined above).

An interesting implication of the finding that $R(t) - L(t)$ is monotonically increasing is that the benefits to effectiveness of diversification over linear combination are greater at larger levels of resource allocation.

COROLLARY 1. *Diversification of identically performing techniques. If techniques i and j perform according to the Exponentially Diminishing Model, and $\lambda_i = \lambda_j$, then $R(t) = X_i R_i(t) + X_j R_j(t)$.*

PROOF. If $\lambda_i = \lambda_j$, then from above, we have $k = 1$

$$\begin{aligned} R(t) - L(t) = & \quad \lambda_j^t(X_i k^t + X_j - k^{X_i t}) \\ = & \quad \lambda_j^t(X_i 1^t + X_j - 1^{X_i t}) \\ = & \quad \lambda_j^t(X_i + X_j - 1) \\ = & \quad 0 \end{aligned}$$

□

Under the conditions that returns are randomly coincident and exponentially diminishing, diversification intuitively neither benefits nor harms portfolio expected effectiveness when the two component techniques have the same expected effectiveness. In the case where the two techniques have different effectiveness, diversification yields improved expected returns over their linear combination.

The following conjecture expands Theorem 2 to portfolios with more than two techniques.

CONJECTURE 1. *If the effectiveness of all techniques i in a portfolio perform according to the Exponentially Diminishing Model, then the expected effectiveness of the portfolio that allocates resources between all techniques i is greater than the expected return of the linear combination of the individual techniques, i.e. $R(t) \geq \sum X_i R_i(t_i)$*

2.3 Maximum Technique Effectiveness

Thus far, we have operated under the assumption that any technique is capable of detecting any defect. Given the goal of proving that portfolio effectiveness is greater than linear combination effectiveness, this assumption has been very conservative; from the moment that technique m can address at most α percent of defects, the only way to address above α percent defects is to combine m with some other technique. Thus, omitting this factor in the proofs above has simplified them while strengthening their results.

Studies have shown that in practice some classes of defects are essentially unfindable by some techniques [18]. A more realistic Exponentially Diminishing Model includes a factor which corresponds to the maximum percentage of defects addressable by that technique. Let α_i represent the maximum percentage of defects addressable by technique i. Then this more Realistic Exponentially Diminishing Model has the form

$$R_i(t) = \alpha_i(1-\lambda_i^t) \ \{0 < \lambda_i < 1, 0 < \alpha_i < 1, t \geq 0\}$$

For example, the static code checker FindBugs looks for specific code idioms that commonly result in defects [4]. Defects not matching one of these idioms are unfindable by FindBugs. Let us say that the percentage of defects that fall into one of the detected patterns, and thus the percentage of defects addressable by FindBugs is 15%, and that FindBugs's DAR on applicable defects is 0.5. Then a Realistic Exponentially Diminishing Model for FindBugs would be $R_i(t) = 0.15(1-0.5^t)$.

2.4 Model Assumptions

This paper's models do not account for fixed overhead in the utilization of techniques. Kan et al. relate a "characteristic S-curve" for the creation of test artifacts over a project's timeline, with a ramp-up period at the beginning of a project and the tailing end of the curve closely resembling an exponentially decreasing line [6]. Some economic analyses have been predicated on the existence of an S-curve and an assumption of a linear relationship between the number of test cases in the S-curve and overall return on quality investment [3, 16]. This is a bad assumption, since the law of diminishing returns implies that the marginal value of later test cases is less than the marginal value of earlier ones.

Additionally, life-cycle factors tend to increase the value of early testing, further diminishing the value of late tests [1]. As it concerns this work, with a long enough project horizon, overhead and ramp-up time become small portions of the cost of quality.

This model does not explicitly consider life-cycle factors, which have large cost-benefit implications [1]. Due to phenomena such as "Defect Propagation" and "Bugterial Infection", it typically costs fewer resources to address a defect early in a project's life-cycle [14, 17]. However, since some techniques are only applied at certain parts of the life-cycle (e.g. in the Waterfall model, requirements review would happen at the beginning of the cycle), a technique's DAR will incorporate information about that technique's life-cycle effects, implicitly factoring life-cycle implications into this model.

We also do not explicitly consider the difference between a technique preventing a defect and a technique discovering a defect once it already exists in the program. Again, we believe this difference in cost can be implicitly reflected in a technique's DAR.

3. VARIABILITY OF EFFECTIVENESS

3.1 Modeling Variability

As in MPT, we will let the variability of effectiveness for a technique or a portfolio be the square root of the variance of its returns. In terms of its component R_i, the variance of

a portfolio of two techniques i and j is:

$$Var(R(t)) = \sum_{i=1}^{N} Var(R_i(X_i t)) + \sum_{i=1}^{N}\sum_{j>1} Var(\omega_{ij}) + 2\sum_{i=1}^{N}\sum_{j>1}^{N} Cov(R_i(X_i t), R_j(X_j t)) + 2\sum_{i=1}^{N}\sum_{j>1}^{N} Cov(\omega_{ij}, R_i(X_i t))$$

This equation becomes unwieldy quickly as the number of techniques increases. Rather than trying to analyze this equation analytically, we simulated $Var(R(t))$ with respect to variable resource allocations, correlation, and coincidence.

3.2 Simulating Variability

We ran three types of simulations, each combining a portfolio of two techniques and each with a different independent variable: correlation, coincidence, and resource allocation. In our simulations, all R_i were represented according to the Realistic Exponentially Diminishing Model. The DAR of each technique was taken to be a random variable with a normal distribution. In addition to being facile to work with, the representation of technique effectiveness via the normal distribution is supported by some empirical evidence [7].

Simulations were performed in the R Programming Environment [13]. These simulations exposed several interesting behaviors of the model:

1. Increasing the total amount of resources available for quality techniques reduced overall variability. This is not surprising, as the quantity of unaddressed defects will always be smaller and have less room to vary at higher resource allocation levels.

2. As expected, decreases in correlation between techniques increased portfolio effectiveness and decreased portfolio variability. Both variability and effectiveness had linear relationships with correlation; variability's relationship with correlation was positive, while effectiveness's relationship with correlation was negative.

3. Also as expected, when coincidence between techniques decreased, portfolio effectiveness increased and portfolio variability decreased. The relationship between coincidence and effectiveness was sigmoidal (Figure 1), as was the relationship between coincidence and variability.

   Most feasible coincidence values clustered around 1.0 times random coincidence. For our model parameters, less than 0.95 times random coincidence or greater than 1.05 times random coincidence produced mostly invalid portfolios (with less than 0 or greater than 1 effectiveness).

   In light of model behavior 2, this suggests that there is a positive and sigmoidal relationship between coincidence and correlation.

4. For any set of model parameters – even those with highly correlated and coincident techniques – it was possible to increase overall resource allocation to a point where the portfolio was at least as effective *and* at most as variable as the linear combination of techniques. Linear combinations had higher effectiveness or lower variability than portfolios only for certain – essentially degenerate – model parameters and only at low resource allocation levels.

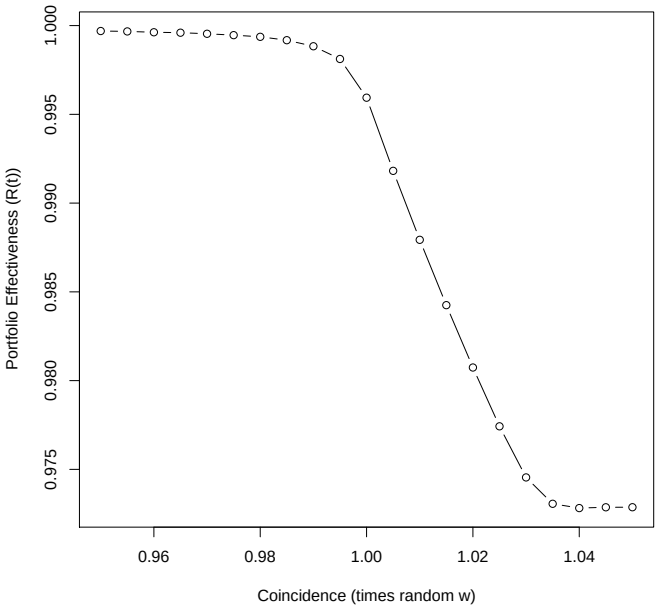

Figure 1: Sigmoidal Relationship between Coincidence and Portfolio Effectiveness

4. MODEL VALIDATION

We analyzed raw data from Wood et al.'s technique combination experiments [18] to test the following hypotheses:

- The effectiveness of individual techniques behaves according to the Exponentially Diminishing Model.
- Combining techniques with negative or small positive correlation in portfolios increases effectiveness and decreases variability.

4.1 Experiment Background

Forty-seven subjects (all students) were asked to find defects in three small C programs using one of three techniques: code reading by stepwise abstraction, functional testing using equivalence partitioning and boundary value analysis, and structural testing using branch coverage. For the remainder of this paper, we shall refer to those techniques as code reading, functional testing, and structural testing, respectively.

Each of the subject programs was about 200 lines long and was taken from a replication package produced by Kamsties and Lott [5]. Two of the programs implemented abstract data types (`ntree` implemented a tree data structure and `nametbl` implemented a symbol table), while the third (`cmdline`) printed the results of parsing a command line. Two of the three programs (`ntree` and `nametbl`) had eight

known defects, while the third (`cmdline`) had nine. Most defects were seeded by Kamsties and Lott; the remainder were introduced during the development of the programs.

In a fractional factorial design, each subject tested each program with a single, different technique for up to three hours. Thanks to the fractional factorial design, no subject tested a program more than once nor utilized a technique more than once.

4.2 Effectiveness and Variability Calculations

For the purposes of this validation, we consider the application of a single technique by a single subject on a program to be a single "unit" of resource. To determine, for example, the average effectiveness of applying one unit of functional testing to program `ntree`, we average the effectiveness of all per-individual results of functional testing `ntree`.

Calculating the average effectiveness of applying more than one unit of resource must consider overlap. For example, to determine the average effectiveness of applying two units of functional testing to `ntree`, we form all possible combinations of two subjects applying functional testing, observe the effectiveness of each pair (avoiding double-counts when both subjects found a given defect), and average effectiveness over all pairs. Table 1 shows an example of this calculation for two fictional subjects and a program with 5 defects.

Subject	Defects Found					Effectiveness
	1	2	3	4	5	
Subject 1	T	F	F	T	F	40%
Subject 2	F	T	F	T	T	60%
Subjects Paired	T	T	F	T	T	80%

Table 1: Example Effectiveness Calculation

To determine variability of effectiveness, each subject's overall performance is taken to be one experiment sample. In the example of Table 1, the variability would be the standard deviation of the effectiveness of the two subjects', 40% and 80%, which is 28%.

To determine the correlation between techniques, the sample is considered to be the per-defect effectiveness of the technique; each defect is a data point used for comparing techniques. For this calculation, effectiveness is a vector, each element of which corresponds to a defect. Vector elements corresponding to addressed defects have value 1, while those not found have value 0.

For example, if our fictional subjects 1 and 2 had used different techniques, we could use their effectiveness data to compute the correlation between techniques. In the example of Table 1, the vector for subject 1 is $(1, 0, 0, 1, 0)$; for subject 2 it is $(0, 1, 0, 1, 1)$. The value of the resulting correlation is -0.16.

4.3 Validating the Exponentially Diminishing Model

Each technique was applied to each program by 14 to 17 subjects. For each technique-program pair, we calculated the average effectiveness of each resource level up to the maximum allocation of 14-17 for that pair. Figure 2 illustrates the results of these calculations for the three techniques against one of the programs (`ntree`).

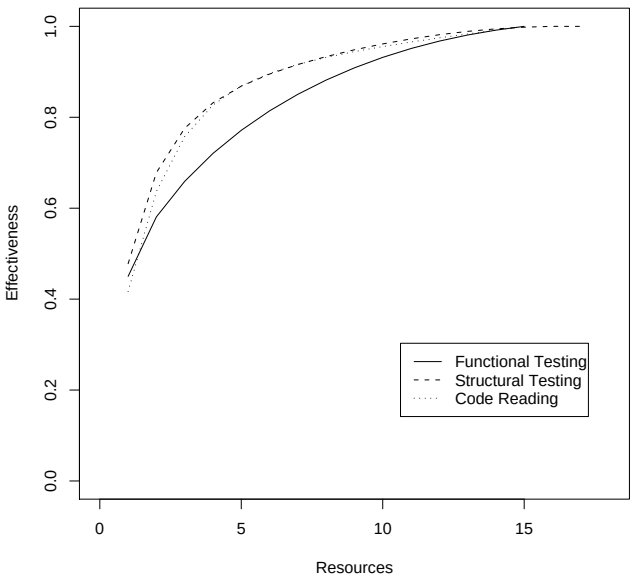

Figure 2: Single Technique Effectiveness on Ntree

To validate the Exponentially Diminishing Model, an exponential regression was performed against this effectiveness data, the results of which are given in Table 2. This nonlinear least squares regression was performed in the R statistical environment [13]. Relatively low residual sum-of-squares (RSS) validate the use of the Exponentially Diminishing Model to model single technique effectiveness.

Technique	Program	DAR	RSS
Structural testing	cmdline	0.69	0.100
	nametbl	0.34	0.047
	ntree	0.61	0.025
Code reading	cmdline	0.74	0.056
	nametbl	0.57	0.009
	ntree	0.63	0.014
Functional testing	cmdline	0.53	0.003
	nametbl	0.37	0.017
	ntree	0.71	0.050

Table 2: Exponential Regression of Single Technique Effectiveness

4.4 Testing the Portfolio Hypothesis

According to the portfolio hypothesis, combining techniques with negative or small positive correlation in portfolios increases effectiveness and decreases variability. To test this, we constructed portfolios of two techniques, varying the allocation of resources between the component techniques. According to the hypothesis, portfolios combining less correlated techniques would see more dramatic and beneficial "efficient frontiers." That is, portfolios with less correlated techniques would have relatively greater effectiveness and relatively lower variability when resources were allocated to both techniques rather than just one or the other.

The analysis of variance of effectiveness Wood et al. performed showed that both the program and the technique were significant sources of variance, and that the interaction between the two was so significant as to prevent separating out the effects on variance between the two [18]. This has been borne out by the relatively high variation between technique DAR's when applied to different programs, as illustrated in Table 2.

Thus, rather than try to test the portfolio effectiveness correlation between techniques across all three programs, we calculated correlation and tested the portfolio hypothesis against each pair of techniques on a per-program basis. Table 3 shows the experimentally derived correlation between each pair of techniques with respect to each program.

Techniques	cmdline	nametbl	ntree
Structural/Functional	0.6998	0.6738	0.5232
Structural/Reading	-0.3157	0.3297	0.6845
Functional/Reading	-0.0947	0.5156	0.7092

Table 3: Correlation of Effectiveness between Techniques

For each pair of techniques and each program, we constructed portfolios of 6 resources, varied the resources allocated between the two techniques, and calculated the effectiveness and variability of the portfolio for each resource allocation. The resource level 6 was heuristically chosen to avoid constructing portfolios that consistently found all the defects (which would have effectively "washed out" the results).

Even this relatively low resource level resulted in a large body of portfolio combinations from which to average effectiveness and calculate variability. For example, there were 14 subjects who applied structural testing to the `cmdline` program, and 17 subjects who applied code reading to it. For the portfolio calculation that allotted 2 resources to structural testing and 4 resources to code reading, we generated all ${}_{14}C_2 = 91$ combinations of 2 subjects structural testing and all ${}_{17}C_4 = 2380$ combinations of 4 subjects reading code. We then took the cross product of these two sets to generate all 216580 portfolios of 2 units of structural testing `cmdline` and 4 units of code reading `cmdline`. The mean percentage of defects found per portfolio and the standard deviation of these percentages form the portfolio effectiveness and variability, respectively, for that portfolio. Figures 3, 4, and 5 show the results of these portfolio calculations for the three programs.

As resources were shifted from all of one technique to all of the other, the relationship between effectiveness and variability fell into one of three categories:

1. Some portfolios saw no trade-offs between effectiveness and variability. These portfolio results ranged from high effectiveness and low variability to low effectiveness with high variability. Portfolios belonging to this category were not beneficial. This comprised of all of the portfolios of code reading and functional testing, the structural testing / functional testing portfolio of `cmdline`, and the structural testing / code reading portfolio of `ntree`. The techniques of these portfolios were highly correlated (0.71, 0.52, 0.70, 0.68), with the exception of functional testing and code reading (-0.09).

2. One set of portfolios exhibited an effectiveness/variability trade-off: the structural/functional portfolios of `nametbl`. Its results ranged from low effectiveness with low variability (all structural testing) to high effectiveness with high variability (all functional testing). The correlation between the two techniques for `nametbl` was relatively high (0.67).

3. Three of the nine sets of portfolios exhibited efficient frontiers where effectiveness was highest from a mixed portfolio, and certain allocations could be chosen to reduce variability at the cost of effectiveness. This was true of the code reading / structural testing portfolios of `cmdline` and `nametbl`, as well as the structural testing / functional testing portfolios of `ntree`. This group tended to be negatively or less positively correlated (-0.32, 0.52, 0.33). The most dramatic efficient frontier belonged to the structural / reading portfolios of `cmdline`, which also had the only large negative correlation of the nine sets of portfolios considered.

   It is interesting to note that each of the programs had exactly one set of portfolios that resulted in an efficient frontier.

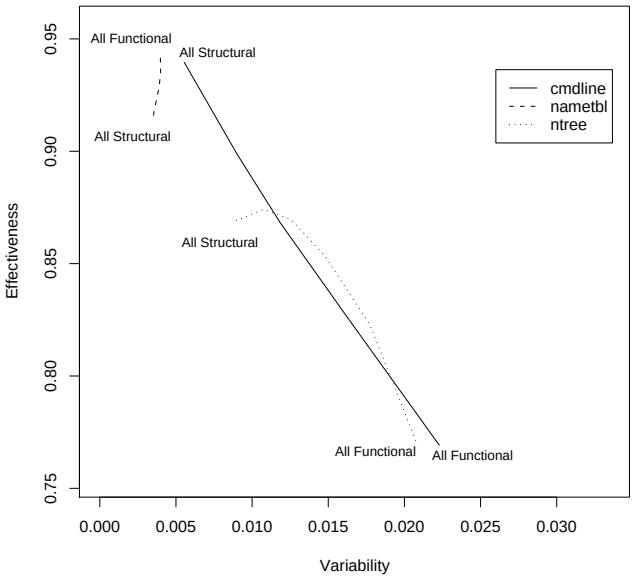

Figure 3: Portfolios of Structural and Functional Testing

Wood et al. analyzed the faults that some techniques addressed much more effectively than others [18]. Their analyses illuminate why some portfolios manifested efficient frontiers while others did not. For instance, fault 3 of `cmdline`, an unused argument that caused failures only under certain conditions, was much more readily detected by code reading than either of the other techniques. (Code reading and structural testing had an efficient frontier for `cmdline`.) Fault 3

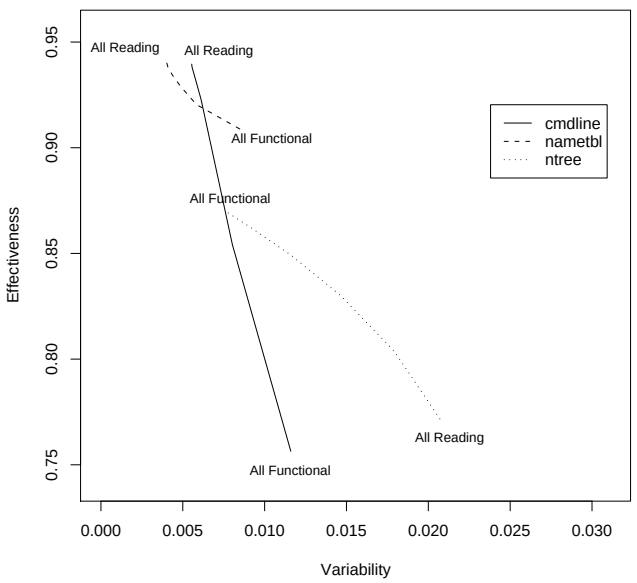

Figure 4: Portfolios of Functional Testing and Code Reading

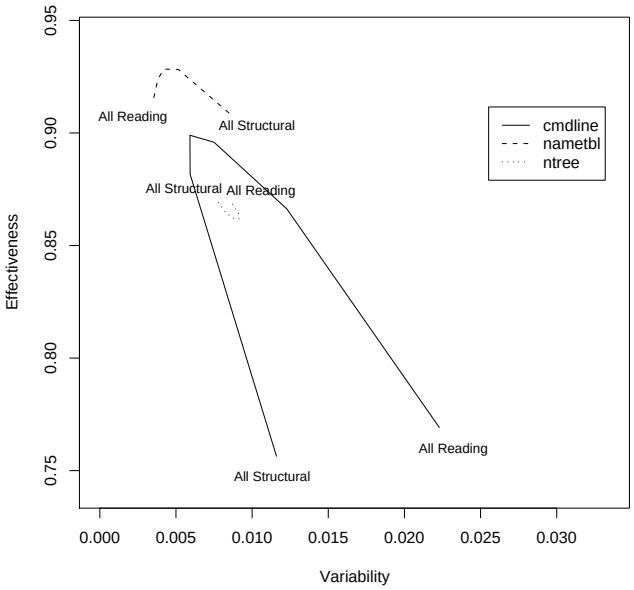

Figure 5: Portfolios of Structural Testing and Code Reading

of `ntree`, a missing `malloc` statement, was effectively found by structural testing but "practically invisible" to functional testing. (Structural testing and functional testing had an efficient frontier for `ntree`.) Overall, most of the faults singled out by Wood et al. for analysis came from programs with portfolio efficient frontiers for the two techniques with divergent performance on that fault.

These results provide some, but by no means definitive, support for the portfolio hypotheses that negative or less positive correlation between component techniques enable portfolios to achieve higher effectiveness and lower variability. The techniques that work best together address different faults and tend to have negative or small positive correlation.

5. CONCLUSION

Our results provide strong evidence that resource allocation to software quality techniques is subject to exponentially diminishing returns. They also indicate that correlation can be used to measure technique complementarity and inform the resource allocation process; combining relatively uncorrelated or negatively correlated techniques tends to decrease variability and increase effectiveness. Although this relationship does not always hold, it holds promise for improving and mitigating the variability of software quality.

Correlation, overlap, and effectiveness are general concepts that should be applicable to quality techniques in other engineering disciplines. We would expect – subject to exponentially diminishing returns – that quality portfolio theory could be used in other fields to positive effect.

More research is needed to understand what factors affect the variability and effectiveness of software quality techniques, and to measure how different techniques correlate. With better understanding of these factors and of the interplay between techniques, we can improve software quality while also making it more controllable and predictable.

6. ACKNOWLEDGMENTS

The author would like to thank Edward Jenvey for his assistance with math and statistics, Alex Gilman and Marat Boshernitsan for their research advice, and Alberto Savoia, Bob Evans, and Venkatesh Prasad Ranganath for their feedback on this paper. Murray Wood and his collaborators have my deepest gratitude for furnishing the raw data used for model validation.

7. REFERENCES

[1] H. Do and G. Rothermel. An empirical study of regression testing techniques incorporating context and lifetime factors and improved cost-benefit models. In *SIGSOFT '06/FSE-14*, pages 141–151, New York, NY, USA, 2006. ACM Press.

[2] L. M. Foreman and S. H. Zweben. A study of the effectiveness of control and data flow testing strategies. *Journal of System Software*, 21(3):215–228, 1993.

[3] G. J. Holzmann. Economics of software verification. In *PASTE '01: Proceedings of the 2001 ACM SIGPLAN-SIGSOFT workshop on Program analysis for software tools and engineering*, pages 80–89, New York, NY, USA, 2001. ACM Press.

[4] D. Hovemeyer and W. Pugh. Finding bugs is easy. *SIGPLAN Notices*, 39(12):92–106, 2004.

[5] E. Kamsties and C. Lott. An empirical evaluation of three defect detection techniques. Technical Report ISERN 95-02, Department of Computer Science, University of Kaiserslauten, 1995.

[6] S. Kan, J. Parrish, and D. Manlove. In-process metrics for software testing. *IBM Systems Journal*, 40(1), 2001.

[7] O. Laitenberger. Studying the effects of code inspection and structural testing on software quality. In *ISSRE '98: Proceedings of the The Ninth International Symposium on Software Reliability Engineering*, page 237, Washington, DC, USA, 1998. IEEE Computer Society.

[8] L. Lauterbach and W. Randall. Experimental evaluation of six test techniques. In *COMPASS '89*, pages 36–41, 1989.

[9] B. Littlewood, P. Popov, L. Strigini, and N. Shryane. Modelling the effects of combining diverse software fault removal techniques. *IEEE Transactions on Software Engineering*, 2000.

[10] H. Markowitz. Portfolio selection. *Journal Of Finance*, 1952.

[11] G. J. Myers. A controlled experiment in program testing and code walkthroughs inspections. *Communications of the ACM*, 21(9), September 1978.

[12] A. Porter and L. Votta. Comparing detection methods for software requirements inspections: A replication using professional subjects. *Empirical Software Engineering: An International Journal*, 3(4):355–379, December 1998.

[13] R Development Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2005. ISBN 3-900051-07-0.

[14] A. Savoia. The developer testing paradox, 2005. http://www.developertesting.com/archives/month200501/20050127-TheDeveloperTestingParadox.html.

[15] R. W. Selby. Combining software testing strategies: An empirical evaluation. In *Proceedings of the Workshop on Software Testing*, July 1986.

[16] S. Wagner. Software quality economics for combining defect-detection techniques. In *Proceedings of Net.ObjectDays*, 2005.

[17] S. Wagner. A model and sensitivity analysis of the quality economics of defect detection techniques. In *ISSTA*, 2006.

[18] M. Wood, M. Roper, A. Brooks, and J. Miller. Comparing and combining software defect detection techniques: a replicated empirical study. In *ESEC '97/FSE-5: Proceedings of the 6th European conference held jointly with the 5th ACM SIGSOFT international symposium on Foundations of software engineering*, pages 262–277, New York, NY, USA, 1997. Springer-Verlag New York, Inc.

APPENDIX

A. SIMULATION PARAMETERS

The simulations presented in Section 3.2 used the following sets of parameters:

1. Varying resources: 100000 samples, 0.0 correlation, 1× random coincidence, 10 total resources to allocate, first technique had average DAR 0.2 and DAR standard deviation 0.3, second technique had average DAR 0.1 and DAR standard deviation 0.001.
2. Varying correlation: 100000 samples, correlation varied between -0.9 and 0.9, 1× random coincidence, each technique allocated 5 resources, first technique had average DAR 0.6 and DAR standard deviation 0.15, second technique had average DAR 0.5 and DAR standard deviation 0.01.
3. Varying coincidence: 100000 samples, 0.0 correlation, coincidence varied between 0.95× and 1.05× random coincidence, each technique allocated 5 resources, first technique had average DAR 0.6 and DAR standard deviation 0.15, second technique had average DAR 0.5 and DAR standard deviation 0.01.