Accuracy and Precision in Finance

30 September 2015

Manfred Gilli,[1] Enrico Schumann[2]

Emilio Fontela's scientific approach was essentially pragmatic; it avoided complex formalisations and focused instead on how the research could contribute to an understanding of the problem under consideration. We are convinced that this essay is in line with the spirit of Emilio Fontela.

M.G.

1 Introduction

Being precise – being clear and unambiguous – is a desirable property of many things: time schedules, cooking recipes, directions given to a traveller, physical measurements. Yet, precision alone is not enough. Time schedules are worth little when the scheduled events do not happen; travel directions are useless when they lead to the wrong place. In other words, just because something is precise does not mean that it is correct.[3] (Even everyday language makes this distinction: one cannot be accurately, but only precisely wrong.)

In this essay, we will use the word *precise* in the meaning of exact and often in the sense of numerically exact. We will use the word *accurate* to indicate that something is both correct and precise enough to be useful. It is the distinction between the two words that we want to discuss. More concretely, we will discuss how researchers and practitioners in finance often mistake precision for accuracy.

To be sure, accuracy and precision are related, and a certain degree of precision is necessary for being accurate. But there must be limits. Ask a runner what the marathon distance is, and the answer will be 42.195 km. But when you think about it, that is a surprisingly precise answer. Why not 42.2 km? That would be just 5 metres off, or 0.01 % of the total distance. Curiously, few people seem to consider this precision strange. But perhaps we have asked the wrong question. Better would have been: what distance does the average marathon participant run? The answer: very likely more than 42.195 km. That is because in professional running, a so-called 'short course prevention factor' is used. From `http://www.bcathletics.org/main/rr_iaaf.htm`:

> 'To prevent a course from being found to be short on future re-measurement, it is recommended that a "short course prevention factor" be built in when laying out the course. For bicycle measurements this factor should be 0.1 % which means that each km on the course will have a "measured length" of 1001 m.'

So, it is certainly part of marathon culture that the marathon distance is 42.195 km. Just as it is part of marathon reality that runners have to make a (slightly) longer distance.

For the marathon runner, the difference between perception and reality – between precision and accuracy – entails little cost. But our thesis is this: the difference between precision and accuracy should receive more attention in finance and economics, because researchers, investor, regulators and other participants in financial markets routinely confuse precision with accuracy, mistaking the former for the latter. And such confusion is costly. Investors may be fooled into overpaying for small advantages, which cannot be replicated in future. They may even fool themselves, for example by being overly reliant on so-called quantitative risk management or by believing that they can fine-tune the risk and reward of their portfolios.

Researchers may give up many opportunities for better research by insisting on precision. In fact, we think that researchers have given up accuracy in favour of precision, and that this is both unwarranted and unfortunate. Instead, because the accuracy of many financial models is low, researchers should give up precision. Through an example, we will illustrate that nothing substantive is lost when alternative, less-precise methods are used. On the contrary, much is gained, since these methods make essentially no assumptions about the data or model and have no requirements when it comes to model specification.

Our essay is not a survey.[4] Instead, we will concentrate on a single topic, investment management. Even

[1] Université de Genève and Swiss Finance Institute; `manfred.gilli@unige.ch`

[2] AQ Investment AG, Zurich; `es@enricoschumann.net`

[3] See Porter (2007) for more on the history of precision in the sciences.

[4] Neither is anything we say really new; much of what we say has been said before: just read Morgenstern (1963). Nevertheless, we feel that it helps to repeat.

more concretely, we will narrow down our discussion on quantitative portfolio management. Managing financial assets is at the heart of financial economics, and hence it is natural to start here. Importantly, as well, it is the area of finance that is most closely related to practical life, and it is here where we feel that ignorance about precision and accuracy can do harm.

Our essay is structured as follows. In the next section, we will discuss models and how errors may enter them. We will focus on one model, that of Markowitz. The key point will be that empirical errors – those resulting from modelling assumptions and parameter inputs – are much greater than numerical errors. That will lead us to heuristics (Section 3) and an illustration of the accuracy of portfolio models (Section 4). Section 5 concludes.

2 Models and Errors

Financial economics, it has been said,[5] is about two questions: (i) how much to save?, and (ii) how to save? It is hard to overstate the importance of these questions, both on an individual and on a macroeconomic level.

When it comes to 'how to save?', it is fair to say that work on an answer began with Harry Markowitz in the 1950s. Markowitz argued that risk should receive much more attention by investors than it had by then, and he argued that the portfolio matters more than single assets. These insights lead him to the one-period mean–variance model.

Over the decades, mean–variance optimisation has developed into the cornerstone of quantitative portfolio management. Yet it remains, of course, a model; it is not the actual problem. The problem is how to save: how to identify assets that give, loosely speaking, much reward with little risk. Markowitz's model is one attempt at solving the problem, by assuming a simple investment process (buy-and-hold) and fixing the notions of reward (mean return) and risk (variance of return).

In general, modelling is the process of putting the actual problem into a form that can be understood by a computer.[6] The modeller has to define vague notions such as 'risk', and often needs to simplify and approximate. This, in turn, means errors. Not in the sense that something went wrong or did not work as expected. Approximation errors originate from the very practice of modelling.

Following a classic discussion in von Neumann and Goldstine (1947), we group these errors into two categories: empirical errors (model errors), and numerical errors. The challenge is to not only acknowledge such errors, but to evaluate them. We are fortunate in finance because we can often measure the magnitude of errors in meaningful units, namely euros, dollars or another currency. Some errors are simply bigger than others and so matter more. Such an evaluation is often case-specific, imprecise and may require interpretation and judgement. But carefully exploring, quantifying and discussing the effects of model choices etc. should always be preferred to dismissing such analysis as 'out of scope'.

Empirical Errors

A portfolio in Markowitz's model is, in essence, a return distribution, which looks good or bad according to an objective function. How we define this objective function determines what portfolio we choose. The word risk for instance is often used synonymously with return variance, but that is by no means the only possible definition. A typical objection against variance is that it penalises upside as well as downside. And indeed, already in the 1950s, Markowitz thought about using downside semi-variance instead, which corresponds much better to the financial practitioner's notion of risk (Markowitz, 1959). To quantify how relevant these differences in model specification are, we need to empirically compare[7] different models with respect to how they help to solve the actual problem.

There are other factors than the objective function that affect the quality of a model. Transactions costs for instance may be relevant. A model that includes them may be better – more accurate – than a model that does not. That does not mean that we should put every detail into the model. Less can be more. Simple back-of-the-envelope calculations may convince us that particular details cannot matter. For instance, modelling a portfolio with actual positions sizes (integers) instead of weights will make a difference only for very small portfolios. It also matters little that we do not know the prices at which we actually open or close positions, which implies that we cannot really know the weights of assets. But more often, whether a certain aspect should be modelled or not is not clear from the start. Thus, we will need to run different models and evaluate and compare their results. In any case, whether

[5] In Constantinides and Malliaris (1995).

[6] A computer is actually not necessary for modelling. Yet, today, it is hard to conceive an investment model that would not rely on a computer.

[7] This essentially means careful data analysis and replication. See, for example, Cohen (1994).

a particular aspect becomes part of the model or not should be motivated from the view point of the actual problem; an aspect should not be neglected because it would make the problem unwieldy or difficult to solve.

Once a model is established, it requires a link to reality, which comes in the form of forecasts and expectations, which often enter as parameters. We can, for instance, only minimise portfolio variance if we have a variance–covariance matrix of the assets' returns. Such model inputs may be good or bad, and we have another source of error. The difficulties in forecasting the required variables are well-described in the literature; see Brandt (2009) for an overview. And it is not only the forecasting problem: results are often extremely sensitive to seemingly minor setup variations, such as the chosen time horizon (LeBaron and Weigend, 1998; Acker and Duck, 2007; Gilli and Schumann, 2010). That makes it difficult to reliably compare models, and hence it becomes difficult to reject bad models.

Numerical Errors

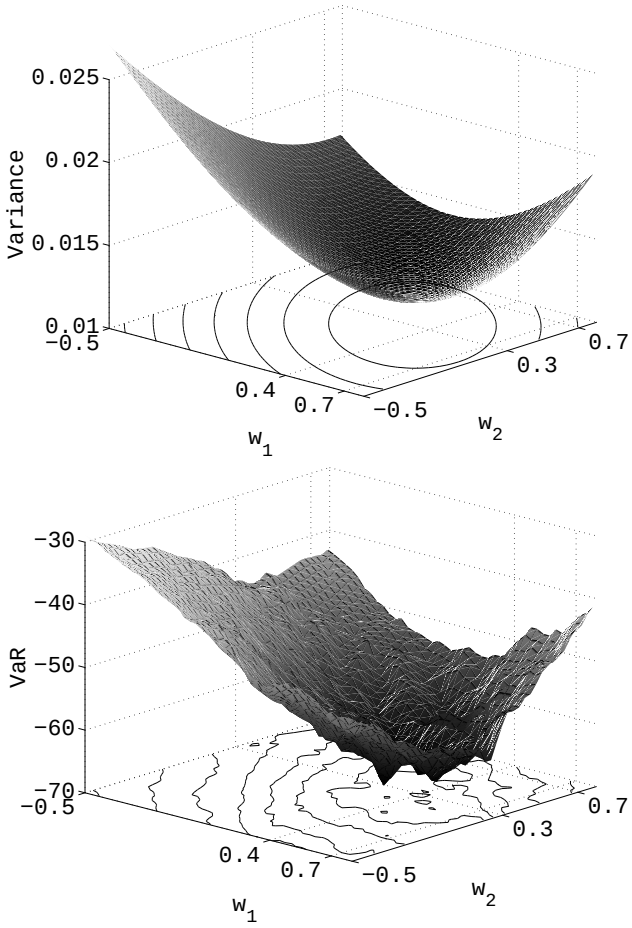
Once the model and its inputs have been fixed, we need to solve it. For this, we use a computer, and with it come two sources of error: round-off error, because we cannot represent all numbers within finite memory; and truncation error, because all computations that 'go to the limit', be it zero or infinity, must stop before the limit is reached.

Round-off error should rarely be a concern in financial applications (see also Trefethen, 2008). It can cause trouble and, more likely, can be a nuisance. But its impact, when compared with the empirical errors described above, is many orders of magnitude smaller.

Truncation error is more relevant to our focus on financial optimisation. In principle, we could solve any optimisation model through random sampling. If we sampled ever more candidate solutions, we would – in principle – come arbitrarily close to the model's solution. But clearly, in most cases that would be an extremely inefficient way to handle a model.

With heuristics, the methods that we want to advocate in this essay and that we describe in the next section, we face a variant of this truncation error. We have not yet explained what heuristics are, but it suffices to say that they are iterative numerical methods for solving optimisation models. The truncation error results because heuristics only provide an approximation to the model's solution. The quality of this approximation is a function of the computational effort we make. With more effort – most easily measured as elapsed computing time –, we obtain better solutions.

Figure 1: Objective function values for a portfolio selection model with three assets. x- and y-axis show weights for two assets; the third weight is fixed through the budget constraint. Upper panel: objective function is variance. Lower panel: objective function is Value-at-Risk.



Of course, only obtaining an approximation of a solution is not satisfactory from the standpoint of optimisation theory. After all, a model's solution is the optimum; theoretically, there are no better or worse solutions, only *the* solution.

Within the context of our discussion, this truncation error that comes with heuristics is simply a lack of precision. Yet imprecise solutions may still be preferred, namely when they belong to more-accurate models – models that would be too difficult to solve precisely. And it turns out that in portfolio selection most models are difficult to solve. As an example, Figure 1 shows, in its upper panel, the variance of a portfolio consisting of three assets. (Actually the square root of the variance. The third asset's weight is fixed through the budget constraint.) This is Markowitz's objective function. In the lower panel we use the same dataset, but this time the objective function is Value-at-Risk, a quantile of the return distribution. The function for Value-at-Risk is

not smooth and a classic method that uses the gradient may become stuck in a local minimum. But Value-at-Risk and similar functions that treat risk asymmetrically are more accurate in the sense that they are conform more closely with the notion of risk. But such models are often rejected simply because they cannot be solved precisely. In finance, many models were written the way they are only because they can be solved precisely. Markowitz himself preferred variance over semi-variance because he could solve the model.[8]

This brings us to heuristics, since they were designed to overcome such local minima, as we will discuss in the next section. However, let us provide a short summary first. The key point that we wanted to make is that selecting a financial portfolio is much more than running an optimisation algorithm. Rather, we move from the actual problem to a model, and from there to the model's solution. (And, of course, we may finally want to implement the model-solution.) During this process, errors will be introduced, but what matters is how large these errors are, and which type – numerical or empirical – matters more. At least on the latter question, we have a clear view: empirical errors – in particular, anything related to data – are much more important than numerical difficulties. In other words, researchers and practitioners should concentrate on empirical errors, not on numerical issues.

3 Good-enough Methods

The upshot of the view that empirical errors matter more than numerical errors is good news for any researcher who ever felt that he lacked the tools to solve a model. Those tools are called heuristics, and we will describe them in this section.

Different people mean different things when they speak of heuristics. In a general sense, a heuristic is a decision rule or *modus operandi* that (i) often helps to solve a problem or to make a good decision, and that (ii) is simple. This is roughly the definition of Pearl (1984), and it is also how computer scientists and programmers use the word: heuristics as simple rules that provide good solutions in many cases. Not perfect, but good enough, and often the best that is available.

[8] Markowitz (1959, ch. 9) compares variance and semi-variance in terms of cost, convenience, familiarity, and desirability; he concludes that variance is superior in terms of cost, convenience, familiarity – but not desirability. To be fair, Markowitz in the 1950s lacked the computing power that alternatives models require; today, we have this computing power.

In the discussion that follows, we will define heuristics in a narrower sense: as a class of numerical methods for solving optimisation models. Such models are typically written as

$$\text{minimise}_x \;\; \phi(x, \text{data})$$

in which ϕ is a scalar-valued function and x is a vector of decision variables. (With a minus in front of ϕ it becomes a maximisation model.) Often there will be further constraints in the model.

We find it helpful to not think in terms of a mathematical description, but a computer programme:

```
solutionQuality = function(x, data).
```

That is, a programme that maps a solution into its quality, given the data. There is no need for a closed-form mathematical description of the function. Indeed, in many applied disciplines there are no closed-form objective functions. The function ϕ could include an experimental setup, with x the chosen treatment and $\phi(x)$ the desirability of its outcome; or evaluating ϕ might require a complicated stochastic simulation, such as an agent-based model.

Several properties, or requirements, describe an optimisation heuristic further (Zanakis and Evans, 1981, Barr et al., 1995, and Winker and Maringer, 2007, list similar criteria):

- The method should result in a 'good' approximation of the true optimum, with 'goodness' measured in computing time or solution quality.
- The method should be robust when we change the model – for instance, when we modify the objective function or add a constraint – and also when we increase the problem size. Results should not vary too much for different parameter settings for the heuristic.
- The technique should be easy to implement.
- Implementation and application of the technique should not require subjective elements.

In a broad sense, we can differentiate between two classes of heuristics, constructive methods and iterative-search methods.

For a constructive method, an algorithm starts with an empty solution and adds components step-by-step; the procedure terminates when it has completed one

solution. An example: a reasonable low-variance equity portfolio of cardinality N can be constructed by (i) obtaining forecasts for the marginal variances of all eligible assets, (ii) sort the assets by forecast variance and (iii) keep the N assets with the lowest forecast variances in the portfolio (equally-weighted); see Schumann (2013).

For iterative-search methods the algorithm moves from solution to solution, that is, a complete existing solution is modified to obtain a new solution. Such a new solution may be quite different from previous ones, as some methods, such as Genetic Algorithms, create new solutions in a rather discontinuous ways. But still, a new solution will share characteristics with its predecessor (if that was not the case, we would be doing random-sampling). In the remainder of this essay, we shall concentrate on iterative-search methods.

Principles
The following pseudocode makes the idea of an iterative method more concrete.

1: generate initial solution x^c
2: while stopping condition not met do
3: create new solution $x^n = N(x^c)$
4: if $A(\phi, x^n, x^c, \ldots)$ then $x^c = x^n$
5: end while
6: return x^c

We start with a solution x^c, very often a random draw. Then, in each iteration, the function N ('neighbour') makes a copy of x^c and modifies this copy; thus, we get a new candidate solution x^n. The function A ('accept') decides whether x^n replaces x^c, typically by comparing the solutions' objective function values. The process repeats until a stopping condition is satisfied; finally, x^c is returned.

This skeleton of an algorithm applies to standard methods, too. In a gradient-based method, for instance, x would be a numeric vector; N would evaluate the gradient at x^c and then move minus the gradient with a specified stepsize; A would evaluate x^c and x^n, and replace x^c only if x^n is better; if not, the search is stopped.

Heuristics use other, often simpler, mechanisms. More specifically, two characteristics will show up in one form or another in most methods. First, heuristics will not insist on the best possible moves. A heuristic may accept a new solution x^n even if it is worse than the current solution. Second, heuristics make use of randomness. For instance, a heuristic may change x^c randomly (instead of locally-optimally as in a Gradient Search). These characteristics make heuristics inefficient for well-behaved models. But for difficult models – for instance, such with many local optima as in Figure 1 –, they enable heuristics to move away from local optima.[9]

As a concrete example, suppose we want to select N assets, equally-weighted, out of a large number of assets, in such a way that the resulting portfolio has a small variance. We assume that we have a forecast for the variance–covariance matrix available. Then a simple method for getting a very good solution to this model is a Local Search. For a Local Search,

- the solution x is a list of the included assets;
- the objective function ϕ is a function that computes the variance forecast for a portfolio x;
- the function N picks one neighbour by randomly removing one asset from the portfolio and adding another one;
- the function A compares $\phi(x^c)$ and $\phi(x^n)$, and if x^n is not worse, accepts it;
- the stopping rule is to quit after a fixed number of iterations.

Note that Local Search is still greedy in a sense since it will not accept a new solution that is worse than the previous one. Thus, if the search arrives at a solution that is better than all its neighbours, it can never move away from it – even if this solution is only a local optimum. Heuristic methods that build on Local Search thus employ additional strategies for escaping such local optima.

And indeed, with a small – but important – variation we arrive at Simulated Annealing (Kirkpatrick et al., 1983). This variation concerns the acceptance rule A: If the new solution is better, accept it, as before. If it is worse, however, do still accept it, but only with a certain probability. This probability in turn depends on the new solution's quality: the worse it is, the lower the probability of being accepted. Also, the probability of acceptance is typically lower in later iterations; that is, the algorithm becomes more select over time.

[9] Because of these mechanisms a heuristic could, in principle, drift farther and farther off a good solution. But practically, that is very unlikely because every heuristic has a bias towards good solutions, for instance by always accepting a better solution, but accepting a worse one only if it is not too bad. Since we repeat this creating of new candidate solutions thousands of times, we can be very certain that the scenario of drifting away from a good solution does practically not occur.

Random Solutions
The most common objection against using heuristics is that because they explicitly rely on random mechanisms, their solutions are also random. It is then difficult, it is argued, to evaluate the quality of a solution. (The discussion in this section builds on Gilli et al., 2011.)

A naïve approach to solving an optimisation model could be this: randomly generate a large number of candidate solutions, evaluate all solutions and keep the best one as the overall solution.

If we repeated this process a second time, our overall solution would probably be a different one. Thus, the solution x we obtain from this sampling strategy is stochastic. The difference between the solution and the true optimum would be a kind of truncation error, since if we sampled more and more, we should in theory come arbitrarily close to the optimum. Importantly, the variability of the solution stems from our numerical technique; it has nothing to do with the error terms that we may have in models to account for uncertainty. Stochastic solutions may occur with non-stochastic methods, too. Think of search spaces like the one in Figure 1. Because of the many local minima, even a a deterministic method such as Gradient Search would result in different solutions when run from different starting points.

We can treat the result of a stochastic algorithm as a random variable with some distribution D. What exactly this result is depends on our setting. We will want to look at the objective function value (ie, the solution quality), but we may also look at the decision variables given by a solution (ie, the portfolio weights). All these quantities of interest we collect in a vector ϱ. The result ϱ_j of a restart j is a random draw from D.

Of course, we do not know what D looks like. But for a given model, there is a simple way to find out – we sample. We run an algorithm a reasonably-large number of times and each time store ϱ_j. From these realisations we compute the empirical distribution function of the ϱ_j, $j = 1, \ldots,$ number-of-restarts as an estimate for D. For a given model or model class, the shape of the distribution D will depend on the chosen method. Some techniques will be more appropriate than others and give less variable and on average better results. The shape of D will typically also depend on the particular settings of the method, in particular the number of iterations – the search time – that we allow for.

Unlike classical optimisation techniques, heuristics can walk away from local minima; they will not necessarily get trapped. So if we let the algorithm search for longer, we can hope to find better solutions. For minimisation problems, when we increase the number of iterations, the mass of D will move to the left and the distribution will become less variable. Ideally, with ever more computing time, D should degenerate into a single point, the global minimum. There exist proofs of this convergence to the global minimum for many heuristic methods (see Gelfand and Mitter, 1985, for Simulated Annealing; Rudolph, 1994, for Genetic Algorithms; Gutjahr, 2000, Stützle and Dorigo, 2002, for Ant Colony Optimisation; Bergh and Engelbrecht, 2006, for Particle Swarm Optimisation).

Unfortunately, these proofs provide little help in practical applications. They often rely on asymptotic arguments, and many such proofs are nonconstructive. Fortunately, we do not need these proofs to make meaningful statements about the performance of specific methods. For a given model class, we can run experiments. Such experiments also help to investigate the sensitivity of the solutions with respect to different parameter settings for the heuristic. Experimental results are of course no proof of the general appropriateness of a method, but they are evidence of how a method performs for a given class of models; often this is all that is needed for practical applications.[10]

4 Evaluating Accuracy

In this section, we will, by way of an example, illustrate two aspects of the preceding discussion. First, that financial models are sensitive, meaning that small changes in assumptions lead to sizeable changes in the results. Second, that the additional randomness that is introduced by using heuristics is minuscule when compared with the effects of model sensitivity.

Minimum Variance
As a model, we use the long-only minimum-variance (MV) portfolio. Ignoring expected returns altogether, as this model does, is equivalent to assuming equal expected returns for all assets. This assumption is motivated by the difficulties of predicting future returns; see Brandt (2009) for an overview of analytic and em-

[10] Our favourite illustration of 'no theoretical proof, but empirical evidence' is Goldbach's conjecture. In its best-known form, it states that every even number greater than two is the sum of two prime numbers. It is a conjecture because to this day, it has not been proved in all generality. But it has been brute-force–tested for even extremely large numbers, and it held up. Thus: one can have empirical evidence that a method works well, even if one cannot prove its optimality.

pirical results. In consequence, if we cannot really control the return of a portfolio, then lowering portfolio risk may still lead to better risk-adjusted performance (Chan et al., 1999). In fact, there is evidence that low-risk stocks even yield higher returns than justified by asset-pricing models (Blitz and van Vliet, 2007). In any case, even a low-risk portfolio is a risky portfolio and should command a risk premium. Altogether there exists convincing empirical evidence that such a purely risk-based approach leads to portfolios that perform well in out-of-sample tests. See, for instance, Chan et al., 1999, for variance minimisation, or Gilli and Schumann, 2011b, for alternative risk functions.

The data set consists of daily prices (adjusted for splits, etc.) of the 30 stocks that comprise the main German stock index, the DAX.[11] The data span the period from January 2004 to mid-September 2015.

A Walkforward with an Exact Solution …
We compute a rolling-window backtest (a 'walkforward') to evaluate the MV strategy. On the last business day of a month, the algorithm computes a MV portfolio, based on historical data of the past year. Weights are constrained to lie between zero and 10%. The resulting portfolio is then held until the next month-end and its performance is recorded. The overall result consists of those one-month out-of-sample periods, chained together.

We solve the optimisation models with a quadratic-programming (QP) solver. Hence, given the data and our assumptions, there is no element of chance in the setup.

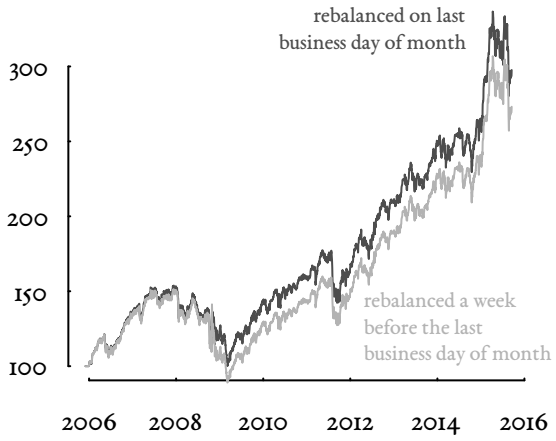
The strategy would have returned a respectable 11.8% per year over the period December 2005 to mid-September 2015. It is pictured in Figure 2.

… but What a Difference a Day Makes
In many publications we would find this performance number to more digits. It is, for instance, unfortunate practice in the industry to compute returns to two decimal places. But such precision is not warranted. To show why, we run the same backtest again, but this time rebalance a week earlier every month. In this test, the overall return drops to 10.8% per year – one whole percentage point less.

Our point is certainly not that rebalancing on the last business day of a month, or on any other particular day of the month, is a good idea. Rather, we wanted to demonstrate what difference a small change in the assumptions of the model can make. This influence of the choice of reference days is documented in the literature (Acker and Duck, 2007); yet, it is widely ignored.

[11] The data and code are available from `http://enricoschumann.net/data/gilli_accuracy.html`.

Figure 2: Performance of MV over the period Dec 2005 to Sep 2015.



Random Windows
To see the effect of the rebalancing days more clearly, we run 10 000 walkforwards with random historical windows and random rebalancing days. The rebalancing dates are randomly spaced between 20 and 80 business days apart; the historical windows span between 120 and 500 business days. The results are shown in Figure 3. The graphic shows a wide variation of outcomes: the 10th quantile of annualised returns is 10.6%, the 90th quantile is 12.3% (the extremes are 8.8% and 15.8%).

Local Search
To demonstrate the effect and magnitude of uncertainty that a heuristic introduces, we re-do the original walkforward. So we rebalance on the last day of the month and use a fixed historical window of 260 business days. But this time, we do not optimise with QP, but with a Local Search, the method we described Section 3. Thus, all the randomness that we may see now comes from our numerical method; there are no other random elements in the setup. Figure 4 shows the results. Compared with the uncertainty introduced by the assumptions on windows, the uncertainty that comes with Local Search is negligible: the grey band that represents the range of outcomes it barely visible.

5 Conclusion (and Some Suggestions)

In this essay we have summarised our view on financial models, their accuracy, and the precision with which

Figure 3: Performance of 10 000 walkforwards with randomly-chosen window sizes. The grey shades indicate different quantiles. Historical windows have a length of between 120 and 500 business days; rebalancing dates are between 20 and 80 business days apart. *All randomness in the results comes through the model and its assumptions; there is no numeric randomness.*

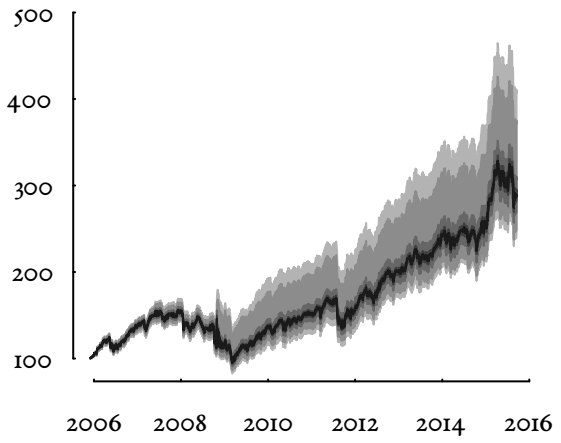


Figure 4: Performance of 10 000 walkforwards with a fixed window size, but performed with a Local Search. As in Figure 3, the grey shades indicate different quantiles; but note that they are barely visible. *All randomness in the results comes through the randomness inherent in Local Search; there are no other elements of chance in the model.*

they are handled.

Our thesis is that researchers and practitioners in finance should pay more attention to accuracy and precision, and, in particular, they should not mistake precision for accuracy. In our view, too much time is spent on making models and their solutions precise in the sense of numerically exact; too little effort is made to really evaluate the accuracy of models.

To some extent, this can be blamed on division of labour. Academics, one might argue, merely develop theoretical models; it is up to the user to properly evaluate them. This argument, however, does not hold water. For one, as we have said above, exploring or just making visible the effects of model choices has always been better than not even attempting any analysis. The good news is that this attitude changes, and in many different disciplines people turn more and more into polymaths rather than narrow specialists, largely because computers and software have become so much better (Gilli and Schumann, 2014).

The lack of accuracy in financial models is good news when it comes to optimisation. Optimisation is an applied discipline; optimisation algorithms are tools. No tool requires 'exact'; 'good enough' is all that is needed (Gilli and Schumann, 2011a). Perversely, a numerically-precise solution to a model may not just not add quality, but also give an unwarranted feeling of being on the safe side. But such a feeling is misplaced: after all, a numerical solution only relates to the model, not to the actual problem.

But then, is there anything we can do to improve financial models and the way they are used? Certainly: analyses should be much more qualitative, relying on non-mechanical insights and research. That means, notably, more exploratory statistics and replication. Findings should be accepted when they are empirically replicated.

That does not imply that single studies do not mean anything. In particular, even a single study can be made more robust. Resampling methods can go a long way to serve as robustness and sensitivity checks.

References

Daniella Acker and Nigel W. Duck. Reference-day risk and the use of monthly returns data. *Journal of Accounting, Auditing and Finance*, 22(4):527–557, 2007.

Richard S. Barr, Bruce L. Golden, James P. Kelly, Mauricio G. C. Resende, and William R. Stewart. Designing and reporting on computational experiments with heuristic methods. *Journal of Heuristics*, 1(1):9–32, 1995.

Frans van den Bergh and Andries P. Engelbrecht. A study of Particle Swarm Optimization particle trajectories. *Information Sciences*, 176(8):937–971, 2006.

David C . Blitz and Pim van Vliet. The volatility effect. *Journal of Portfolio Management*, 34(1):120–113, 2007.

Michael W. Brandt. Portfolio choice problems. In Yacine Aït-Sahalia and Lars P. Hansen, editors, *Handbook of Financial Econometrics*, volume 1. Elsevier, 2009.

Louis K. C. Chan, Jason Karceski, and Josef Lakonishok. On portfolio optimization: Forecasting covariances and choosing the risk model. *Review of Financial Studies*, 12 (5):937–974, 1999.

Jacob Cohen. The earth is round ($p < .05$). *American Psychologist*, 49(12):997–1003, 1994.

George M. Constantinides and Anastasios G. Malliaris. Portfolio theory. In R.A. Jarrow, V. Maksimovic, and W.T. Ziemba, editors, *Finance (Handbooks in Operations Research and Management Science)*, volume 9, pages 1–30. North-Holland, 1995.

Saul B. Gelfand and Sanjoy K. Mitter. Analysis of Simulated Annealing for Optimization. Technical Report LIDS-P-1494, MIT, 1985.

Manfred Gilli and Enrico Schumann. Optimization in financial engineering – an essay on 'good' solutions and misplaced exactitude. *Journal of Financial Transformation*, 28:117–122, 2010.

Manfred Gilli and Enrico Schumann. Optimal enough? *Journal of Heuristics*, 17(4):373–387, 2011a.

Manfred Gilli and Enrico Schumann. Risk–reward optimisation for long-run investors: an empirical analysis. *European Actuarial Journal*, 1(1, Supplement 2):303–327, 2011b.

Manfred Gilli and Enrico Schumann. Optimization cultures. *Wiley Interdisciplinary Reviews: Computational Statistics*, 5(6):352–358, 2014. doi: 10.1002/wics.1312.

Manfred Gilli, Dietmar Maringer, and Enrico Schumann. *Numerical Methods and Optimization in Finance*. Elsevier/Academic Press, 2011.

Walter J. Gutjahr. A graph-based Ant System and its convergence. *Future Generation Computer Systems*, 16(9): 873–888, 2000.

Scott Kirkpatrick, C. Daniel Gelatt, and Mario P. Vecchi. Optimization by Simulated Annealing. *Science*, 220(4598): 671–680, 1983.

Blake LeBaron and Andreas S. Weigend. A bootstrap evaluation of the effect of data splitting on financial time series. *IEEE Transactions on Neural Networks*, 9(1):213–220, 1998. available from `citeseer.ist.psu.edu/lebaron98bootstrap.html`.

Harry M. Markowitz. *Portfolio Selection*. Wiley, New York, 1959.

Oskar Morgenstern. *On the Accuracy of Economic Observations*. Princeton University Press, 2nd edition, 1963.

Judea Pearl. *Heuristics*. Addison-Wesley, 1984.

Theodore M. Porter. Precision. In Marcel Boumans, editor, *Measurement in Economics – A Handbook*. Academic Press, 2007.

Günter Rudolph. Convergence analysis of canonical genetic algorithms. *IEEE Transactions on Neural Networks*, 5(1): 96–101, 1994.

Enrico Schumann. Take-the-best in portfolio selection. available from `http://enricoschumann.net`, 2013.

Thomas Stützle and Marco Dorigo. A short convergence proof for a class of Ant Colony Optimization algorithms. *IEEE Transactions On Evolutionary Computation*, 6(4): 358–365, 2002.

Lloyd N. Trefethen. Numerical analysis. In Timothy Gowers, June Barrow-Green, and Imre Leader, editors, *Princeton Companion to Mathematics*. Princeton University Press, 2008.

John von Neumann and Herman H. Goldstine. Numerical inverting of matrices of high order. *Bulletin of the American Mathematical Society*, 53(11):1021–1099, 1947.

Peter Winker and Dietmar Maringer. The Threshold Accepting optimisation algorithm in economics and statistics. In Erricos John Kontoghiorghes and Cristian Gatu, editors, *Optimisation, Econometric and Financial Analysis*, volume 9 of *Advances in Computational Management Science*, pages 107–125. Springer, 2007.

Stelios H. Zanakis and James R. Evans. Heuristic "optimization": Why, when, and how to use it. *Interfaces*, 11(5): 84–91, 1981.