COMPUTER-INTENSIVE STATISTICAL METHODS

B. D. Ripley
Department of Statistics, University of Oxford, UK

Keywords: Bootstrap, Monte Carlo, Smoothing, Neural network, Classification tree

Contents

Summary

Computer-intensive statistical methods may suitably be defined as those which make use of repeating variants on simpler calculations to obtain a more illuminating or more accurate analysis. Many calculations that would have been unreasonably time-consuming fifteen years ago can now be handled on a standard desktop PC. A common theme is the desire to relax assumptions, for example by replacing analytical approximations by computational ones, or replacing analytical optimization or integration by numerical methods. Many of methods developed recently rely on simulation, repeating a simple analysis many (often thousands) of times on different datasets to obtain some better idea of the uncertainty in the results of a standard analysis. This chapter discusses: resampling and Monte Carlo methods; numerical optimization and integration; density estimation and smoothing; and methods that try to improve on least squares regression and/or handle non-linearity. In this final category are models that work with smooth functions of predictors, neural networks, support vector machines, classification and regression trees, and methods that combine the predictions from multiple models.

1. Introduction

The meaning of ‘computer-intensive’ statistics changes over time: Moore’s Law (whose colloquial form is that there is a doubling of computing performance every 18 months)

means that what was computationally prohibitive 15 years ago is now a task for a small number of hours on a standard desktop PC. One of the things which has changed most about computing is that nowadays the leading supercomputers perform calculations not much faster than a desktop PC, but obtain their power through the ability to do many calculations in parallel. So a good current definition would be

Computer-intensive statistical methods are those which make use of repeating variants on simpler calculations to obtain a more illuminating or more accurate analysis.

A common theme is the desire to relax assumptions, for example by replacing analytical approximations by computational ones, or replacing analytical optimization or integration by numerical methods.

Many of the methods developed recently rely on simulation, repeating a simple analysis many (often thousands) of times on different datasets to obtain some better idea of the uncertainty in the results of a standard analysis. Fortunately, at last most mathematical and statistical packages provide reasonable facilities for simulation, but there is still some legacy of the poor methods of random-number generation which were widespread in the last half of the 20th century.

2. Resampling and Monte Carlo Methods

The best-known simulation-based method is what Efron called the *bootstrap*. To illustrate the idea, consider the simplest possible example, an independent identically distributed sample $x_1,\ldots,x_n$ from a single-parameter family of distributions $\{F(x,\theta)\}$ and an estimator $\hat{\theta}$ of θ. We are interested in the sampling properties of $\hat{\theta}$, that is the variability of $\hat{\theta}$ about θ. Unfortunately, we only have one sample, and hence only one $\hat{\theta}$, but can we '*manufacture*' more samples. Two ways spring to mind:

1. Draw a sample of size n from $F(x,\hat{\theta})$ or
2. Draw a sample of size n from the empirical distribution F_n of $x_1,\ldots,x_n$.

In each case we can compute a new estimate $\hat{\theta}^*$ from the new sample, and use the variability of $\hat{\theta}^*$ about $\hat{\theta}$ as a proxy for the variability of $\hat{\theta}$ about θ. Since we can draw B such samples, and that given enough computing power B could be large, we can explore in detail the variability of $\hat{\theta}^*$ about $\hat{\theta}$: the issue is 'only' how good a proxy this is for the distribution we are really interested in.

2.1. The Bootstrap

The second possibility does not even require us to know F, and is what is known as (non-parametric) *bootstrapping*: the first is sometimes known as the parametric bootstrap. The name comes from the phrase

"pull oneself up by one's bootstraps"

which is usually attributed to the fictional adventures of Baron Munchausen. Sampling

from F_n amounts to choosing independently n of the data points with replacement, so almost all re-samples will contain ties and omit some of the data points: on average only about $1-1/e \approx 63\%$ of the original points will be included.

The bootstrap paradigm is that the proxy distribution provides a good approximation. Bootstrapping has been embraced with great enthusiasm by some authors, but does have quite restricted application. Like many of the techniques discussed in this chapter, it is easy to apply but can be hard to demonstrate the validity.

If we accept the paradigm, what can we do with the proxy samples? We can explore aspects such as the bias and standard error of $\hat{\theta}^*$, and hence replace asymptotic distribution theory by something that we hope is more accurate in small samples. Much research has been devoted to finding confidence intervals for θ. Suppose we want a level $1-\alpha$ (e.g. 95%) confidence interval, and let $k_{\alpha/2}$ and $k_{1-\alpha/2}$ be the corresponding percentiles of the empirical distribution of $\hat{\theta}^*$. The *percentile* confidence interval is $(k_{\alpha/2}, k_{1-\alpha/2})$. The *basic* confidence interval is $(2\hat{\theta}-k_{1-\alpha/2}, 2\hat{\theta}-k_{\alpha/2})$, that is the percentile CI reflected about the estimate $\hat{\theta}$. (The two are frequently confused.) The advantage of the percentile distribution is that it transforms as one would expect, so that taking the percentile interval for $\phi = \log\theta$ is the log of percentile interval for θ, *but* if $\hat{\theta}$ is biased upwards, the percentile interval will be doubly biased upwards.

There are several ways to (possibly) improve upon the basic and percentile intervals. Both BC_a intervals and the double bootstrap use intervals $(k_{\beta_l}, k_{\beta_u})$ and choose the β's appropriately, in the case of the double bootstrap by a second layer of bootstrapping. As the chosen percentiles tend to be quite extreme, these methods often need many bootstrap re-samples and can be seriously computationally intensive even with the resources available in 2004.

It is less easy to apply bootstrapping to more structured sets of data. Suppose we have a regression of n cases of y on $\boldsymbol{x}$. It may be possible to regard the n cases $(\boldsymbol{x}, y)$ as a random sample and apply simple bootstrapping to cases. However, if this were the result of a designed experiment we do want to cover the whole set of $\boldsymbol{x}$ values chosen, and even for an observational study we usually want to estimate the variability conditional on the $\boldsymbol{x}$s actually observed. One idea is to resample the residuals and create new samples treating these as 'errors': the various approaches can lead to quite different conclusions. Bootstrapping time series or spatial data is trickier still.

2.2. Monte Carlo Methods

The other possibility, to simulate new datasets from the model, makes most sense in a significance testing situation. Suppose we have a simple null hypothesis $H_0 : \theta = \theta_0$. Then we can simulate m samples from H_0 and get new estimates $\hat{\theta}_i$ from those samples. Then under H_0 we have $m+1$ samples from $F(,\theta_0)$, the data and the m we generated. Suppose we have a test statistic $T(\theta)$, large values of which indicate a

departure from the null hypothesis. We could compare $T(\hat{\theta})$ to the empirical distribution of the $T(\hat{\theta}_i)$ as an approximation to the null-hypothesis distribution of T, that is to compute $\hat{p} = \#\{i : T(\hat{\theta}_i) > T(\hat{\theta})\}/m$. However, Monte Carlo tests use a clever variation: a simple counting argument shows that

$$P(T(\hat{\theta}) \text{ is amongst the } r \text{ largest}) = \frac{r}{m+1}.$$

Thus we can obtain an *exact* 5% test by taking $r = 1, m = 19$ or $r = 5, m = 99$ or $r = 25, m = 499, \ldots$.

This example has many of the key features of computer-intensive methods: it makes use of a simple calculation repeated many times, it relaxes the distributional assumptions needed for analytical results, and it is in principle exact given an infinite amount of computation. Rather than considering large amounts of data, we consider large amounts of computation, as the ratio of cost of computation to data collection is continually falling.

The simulation-based methods are only feasible if we have a way to simulate from the assumed model. In highly-structured situations we can find that everything depends on everything else. This was first encountered in statistical physics (Metropolis *et al.*) and spatial statistics (Ripley, Geman & Geman). Those authors devised iterative algorithms that only used the conditional distributions of small groups of random variables given the rest. As successive samples are not independent but form a Markov chain (on a very large state space) these methods are known as MCMC, short for Markov Chain Monte Carlo. This is now becoming the most commonly used methodology for applied Bayesian statistics.

3. Numerical Optimization and Integration

A simplistic view of statistical methods is that they reduce to either the optimization or the integration of some function, with Bayesian methods majoring on integration. For reasonably realistic models numerical integration is often (extremely) computer-intensive. Simulation provides a very simple way to perform an integration such as $\phi = Ef(X)$: just generate m samples $X_1, \ldots, X_m$ from the distribution of X and report the average of $f(X_i)$. It is not usually a good way to find an accurate estimate of ϕ, for the central limit theorem (if applicable) suggests that the average error decreases at rate $1/\sqrt{m}$. Nevertheless, this is the main use of MCMC, to obtain a series of nearly-independent samples from a very high-dimensional joint distribution and then integrate out all but a few dimensions just by making f depend on a small number of variables (often just one).

There are competing methods of integration. In a moderate number of dimensions it may be better to use non-independent samples X_i designed to fill the sample space more evenly, sometimes called *quasi-Monte Carlo*.

Numerical optimization can also be computer-intensive, particularly when there are constraints on the parameters. Considerable progress has been made in recent years and it is worthwhile to seek out state-of-the-art software for numerical optimization.

Several areas of modern statistics combine both integration and optimization via maximum likelihood estimation of models with latent variables. Two classic examples are factor analysis and linear mixed effects models. In each case the integration can be performed numerically in the classic cases with normally-distributed latent variables, but the optimization is often challenging. However, if we consider discrete rather than continuous observations such as generalized linear mixed models the integration has to be performed numerically and we can easily find ourselves numerically optimizing a likelihood each evaluation of which involves many high-dimensional integrations, resulting in weeks of computation to fit a single model.

-
-
-

Bibliography

Davison, A. C. and Hinkley, D. V. (1997). *Bootstrap Methods and Their Application*. Cambridge University Press, Cambridge. [A comprehensive and well-balanced account. It includes extensive examples, and includes both the strengths and weaknesses of this approach.]

Evans, M. and Swartz, T. (2000). *Approximating Integrals via Monte Carlo and Deterministic Methods*. Oxford University Press, Oxford. [An accessible account of numerical integration.]

Nocedal, J. and Wright, S. J. (1999). *Numerical Optimization*. Springer-Verlag, New York. [A survey of modern-day methods.]

Pinheiro, J. C. and Bates, D. M. (2000). *Mixed-Effects Models in S and S-PLUS*. Springer-Verlag, New York. [Mixed models are an important application of integration and optimization. This describes the implementation of one approach, and shows the complexity that normally lies behind translating computer-intensive methods into software.]

Ripley, B. D. (1987). *Stochastic Simulation*. Wiley, New York. [The classic reference which even then covered Markov Chain Monte Carlo.]

Ripley, B. D. (1996). *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge. [Views neural networks, classification trees and other non-linear methods as statistical methods. Also covers many parts of smoothing.]

Rousseeuw, P. J. and Leroy, A. M. (1987) *Robust Regression and Outlier Detection*. John Wiley and Sons, New York. [The most accessible account available of robust regression.]

Ruppert, D., Wand, M. P., and Carroll, R. J. (2003). *Semiparametric Regression*. Cambridge University Press, Cambridge. [A modern practically-oriented account of smoothing methods and the ways in which they can be used to relax assumptions in regression problems.]

Simonoff, J. S. (1996). *Smoothing Methods in Statistics*. Springer-Verlag, New York. [The only reasonably comprehensive text on density estimation and smoothing, but packs in many methods and is mathematically sophisticated. Good as a reference resource.]

Venables, W. N. and Ripley, B. D. (2002). *Modern Applied Statistics with S*. Fourth edition. Springer-Verlag, New York. [A graduate-level text on many of the methods described here, with many examples of their use in S, the most widely used computer language for computer-intensive statistics.]

Biographical Sketch

Brian Ripley has held the Chair of Applied Statistics at the University of Oxford since 1990. His Ph.D. from the University of Cambridge was in stochastic geometry and spatial statistics and he subsequently worked in (and published monographs on) spatial statistics, simulation, image analysis, pattern recognition and neural networks. He is nowadays best known for his work in statistical computing, including contributions to the S language and the R project, as well as innovative ways to use simulation and computer power to do better applied statistics.