

Chapter 5

PERFORMANCE ANALYSIS OF FUNCTION OPTIMIZERS

5.1 Introduction

In the last two chapters we have developed a class of genetic plans which performs well on the environment E in comparison with random search. In this chapter we provide an alternate point of comparison by evaluating the performance of several function optimization techniques on E.

In our discussion of function optimization in chapter 1, we noted that the problem of finding function extrema has generally been divided into two subproblems: finding the nearest local extremum (local or unimodal search) and finding the global extremum (global or multimodal search). Many sophisticated techniques have been developed which solve the local search problem (see, for example, Jacoby and Kowalik(1972) or Huang(1970)). However, much less success is evident for the global search problem. Several approaches have been proposed if appropriate bounds can be assumed on the derivatives of the functions to be optimized (see, for example, Bremermann (1970) or Brent (1971)). Alternatively, one can perform some sort of patterned search in an attempt to locate the global optimum (see, for example, Hill (1969)). Unfortunately, for most of these techniques the computation time grows rapidly with the dimensionality

of the problem, and they are used in practice only for low-dimensional problems. As a consequence, one is left with two alternatives for a more general global function optimizer. Either one runs a good local optimizer a sufficiently large number of times to assure all local optima have been found or revert to some form of random search. Since we know we can do better than random search with the genetic algorithms, we consider in this chapter the alternative of restarting a local optimizer.

5.2 Local Optimization Techniques

The most successful local minimization techniques have come from the area of iterative descent methods. The idea here is to reduce the problem of finding the minimum to a sequence of one-dimensional searches along a direction vector p_k . That is, at each step a point x_{k+1} is generated where $x_{k+1} = x_k + \lambda_k p_k$ and $f(x_{k+1}) < f(x_k)$. The techniques differ in how the direction vector p_k and the step size λ_k are chosen. We will consider two different techniques, one which requires that derivative information be available for the function being optimized and one which does not.

A well-studied approach to the choice of direction vectors is to perform a sequence of one-dimensional minimizations along a sequence of conjugate directions. If the function to be minimized is quadratic of dimension n , then in theory only n such one-dimensional

minimizations are required for convergence. In practice, however, conjugate direction techniques are applied to arbitrary functions with heuristic modifications to prevent the conjugate directions from becoming linearly dependent and reducing the search space. Powell (1964) proposed a technique for calculating conjugate directions without derivative information by means of a series of one-dimensional minimizations. Subsequently, Brent (1971) and others have made modifications to improve the performance of this approach. Since Brent's algorithm is available in software form (PRAXIS), it seemed a reasonable choice as a representative of conjugate direction methods which do not require derivatives.

A somewhat more sophisticated approach to the problem of local minimization, using variable metric methods, was introduced by Fletcher and Powell (1963) with subsequent variations proposed by Broyden (1970) and Huang (1970). In this case a sequence of $n \times n$ "metric" matrices (where n is the dimensionality of the search space) are constructed using gradient information to simultaneously provide a linear transformation of the search space into one less badly scaled and provide a sequence of conjugate directions for one-dimensional searches. As a consequence, each step is computationally more expensive than, for example, the previous approach (particularly when n is large), but generally requires no more than n steps on quadratic functions even when

the function to be minimized is badly scaled. In practice, variable metric methods are applied to arbitrary functions with heuristics for preventing the metric matrices H_k from becoming singular. Since the Fletcher-Powell algorithm is available in software form (DFP), it seemed a reasonable choice as a representative of the variable metric methods.

5.3 Performance Evaluation Conventions

One of the most difficult things to find in the function optimization literature is a comprehensive comparative analysis of the performance of various optimization techniques. Invariably, a paper will cite as performance evidence the fact that one technique required fewer function evaluations to minimize a particular function from a particular starting point. In reality one finds that the comparison depends not only on the starting point but also on a number of "hidden" parameters such as one-dimensional search accuracies, initial step sizes, estimates of scaling, and so on. Applying the algorithm to a different function or even a different starting point requires more parameter "tuning" for the published results. In this thesis we have been concerned with the quality of robustness, that is, the ability of an algorithm to perform well in a wide variety of situations. For function optimization analysis, this suggests that we evaluate algorithms

over a variety of starting points and require that any "hidden" parameters be fixed over the duration of the evaluation.

In order to provide direct comparisons with the performance of the genetic algorithms on E, several conventions were adopted. On-line and off-line performance measurements were made for PRAXIS and DFP in each of two modes: local mode and global mode. In local mode, a random starting point is chosen and the algorithm is run until it converges. If convergence occurs within 6000 trials (the interval of observation), the performance measures are extrapolated from the point of convergence out to 6000 trials by assuming $f_e(t) = f_e^*(t) = \widetilde{FMIN}$ where \widetilde{FMIN} is the minimum at convergence. By averaging this performance over a number of random starting points, we have a direct comparison between local optimizers and genetic plans. In global mode, the algorithm is restarted with a new random starting point each time it converges until it has allocated 6000 trials. By averaging global performance over a number of random initial starting points, we have a direct comparison between proposed global optimizers and genetic plans.

Recall from appendix A that each test function in E was in fact restricted to a bounded subspace of R^n of the form $|x_j| \leq b$. PRAXIS and DFP are unconstrained minimization techniques. No attempt was made to prevent excursions outside of the bounded search space. How-

ever, all random starting points were chosen from a uniform distribution over the bounded search space. Recall also that the bounded spaces were discretized by specifying a resolution factor Δx_1 . For fairness in comparison, no finer resolution of the minimum was required of PRAXIS and DFP. For PRAXIS convergence is assumed when successive estimates of the minimum essentially satisfy

$$|x_k - x_{k-1}| \leq T$$

For each test function in E, T was set to the discretization factor Δx_1 . For DFP, convergence is assumed whenever the gradient at a particular estimate of the minimum x_k satisfies

$$G(x_k) \leq \epsilon$$

For each test function in E, ϵ was set to $G(x_{\min} + \Delta x_1)$, the gradient one resolution step away from the minimum.

Finally, an attempt was made to equalize the fact that DFP required derivative information about the functions being minimized. As we noted in chapter 1, in many adaptive system applications, the performance functions to be optimized are not available in mathematical closed forms. Rather, they are usually "black box" problems for which only function values are readily available. This means that derivative information must be estimated by function evaluations taken small distances away. This suggests that gradient information

is equivalent at the very least to n function evaluations (where n is the dimension of the space) and perhaps $2n$ or more depending on the accuracy required. Since the DFP algorithm we used expected exact derivative information, it seemed unfair to use derivative estimates. Rather, exact derivatives were computed upon request for each of the test functions in E, but at the same time n function evaluations were computed as a conservative way of equalizing for this additional information.

5.4 Performance Evaluation of PRAXIS and DFP

Figures 5.1 - 5.10 give the off-line and on-line performance curves produced by PRAXIS and DFP in local mode. Each of these curves represents the average of 20 independent trials using random starting points within the bounded subspaces defined in appendix A.

Figures 5.1 and 5.2 illustrate the performance of PRAXIS and DFP on test function F1. Notice the time scale here in relationship to those of the genetic algorithms in the previous chapter. Both PRAXIS and DFP converge within 60 trials while the genetic algorithms required several thousand. These curves indicate the kind of performance which is possible with local optimizers when the assumptions about the function being minimized actually hold. With a low-dimensional, nicely scaled quadratic function like F1, one can hardly do better. The on-line performance curves on F1 bring

FIG 5.1: OFF-LINE PERFORMANCE ON F1

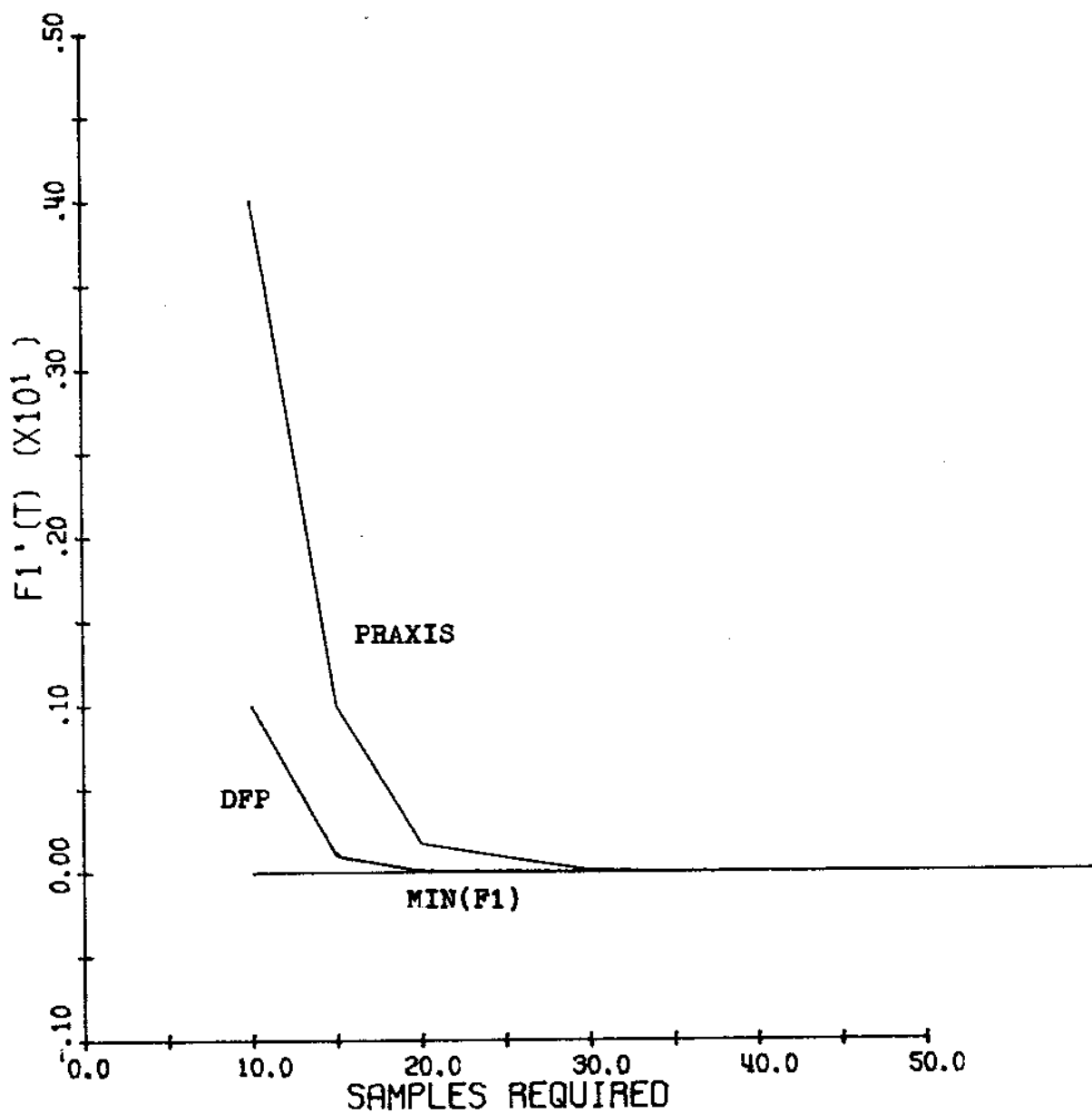


Figure 5.1: Off-line performance curves for PRAXIS and DFP in local mode on F1.

FIG. 5.2: ON-LINE PERFORMANCE ON F1

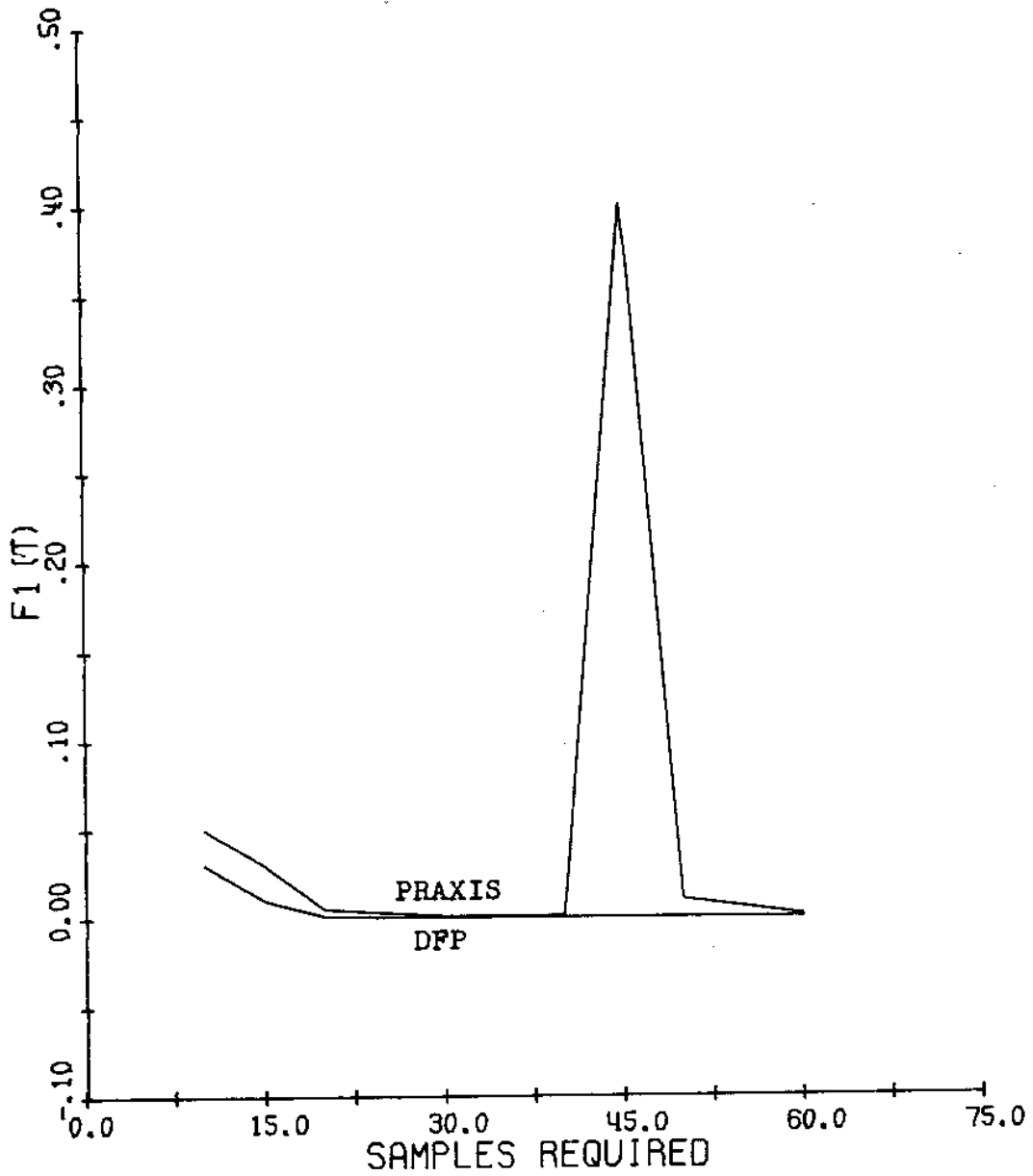


Figure 5.2: On-line performance curves for PRAXIS and DFP in local mode on F1.

out an interesting characteristic of PRAXIS. To avoid the problem of being caught in a narrow valley, when convergence is imminent, PRAXIS tries several steps in random directions. This shows up immediately in on-line performance since the probability of improvement is small.

Figures 5.3 and 5.4 illustrate the local performance curves generated on F2. F2 violates several of the assumptions made concerning the function to be minimized. It is non-convex and non-quadratic, and is also badly scaled. Again, both DFP and PRAXIS converged in far less time than the genetic algorithms, although they both required considerably more trials than on F1. Notice again how the random strategy of PRAXIS shows up in the final stages of on-line performance.

Figures 5.5 and 5.6 illustrate the local performance curves generated on F3. As they indicate, F3 gave both local optimizers considerably more difficulty than F1 and F2. The stopping criterion used by PRAXIS was never satisfied within 6000 trials, requiring manual termination. On the other hand, because DFP used a gradient stopping criterion, it stopped almost immediately on whatever plateau was selected by the random starting point. As a consequence, in local mode it converged on the average to $AVE(F3) = -2.5$, which was then extrapolated as discussed earlier over the remainder of the 6000 trials.

Figures 5.7 and 5.8 illustrate the local performance

FIG. 5.3: OFF-LINE PERFORMANCE ON F2

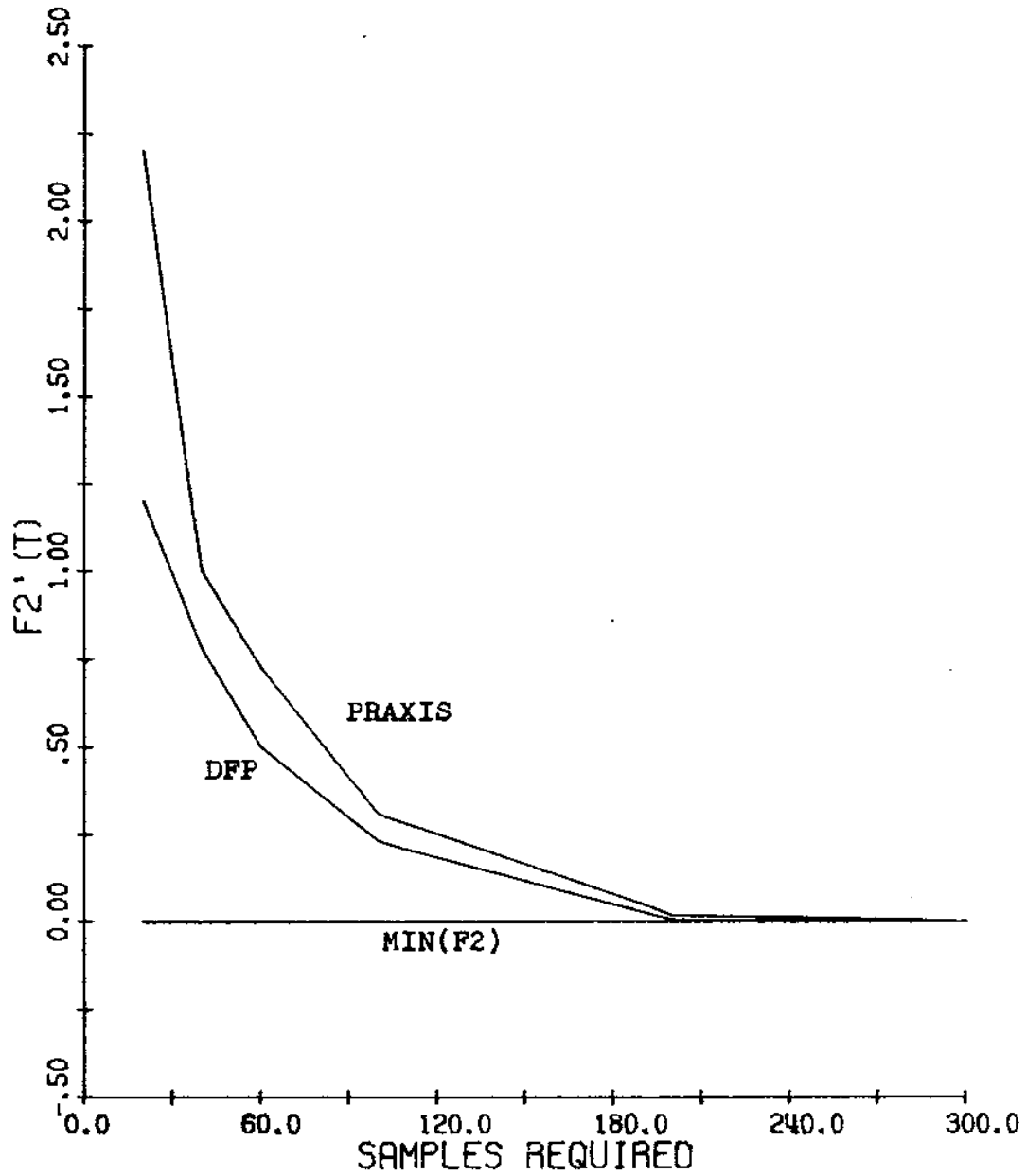


Figure 5.3: Off-line performance curves for PRAXIS and DFP in local mode on F2.

FIG. 5.4: ON-LINE PERFORMANCE ON F2

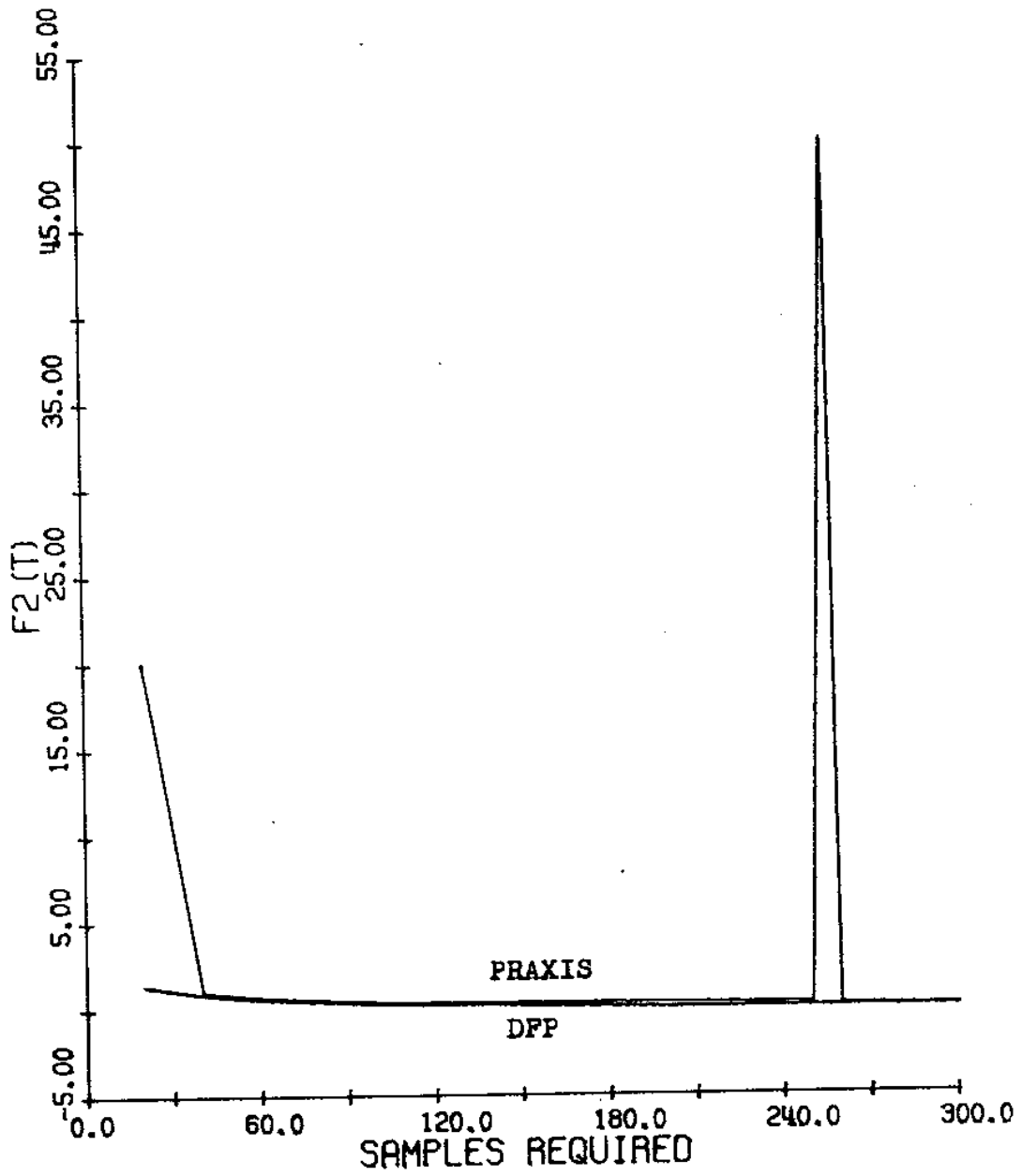


Figure 5.4: On-line performance curves for PRAXIS and DFP in local mode on F2.

FIG 5.5: OFF-LINE PERFORMANCE ON F3

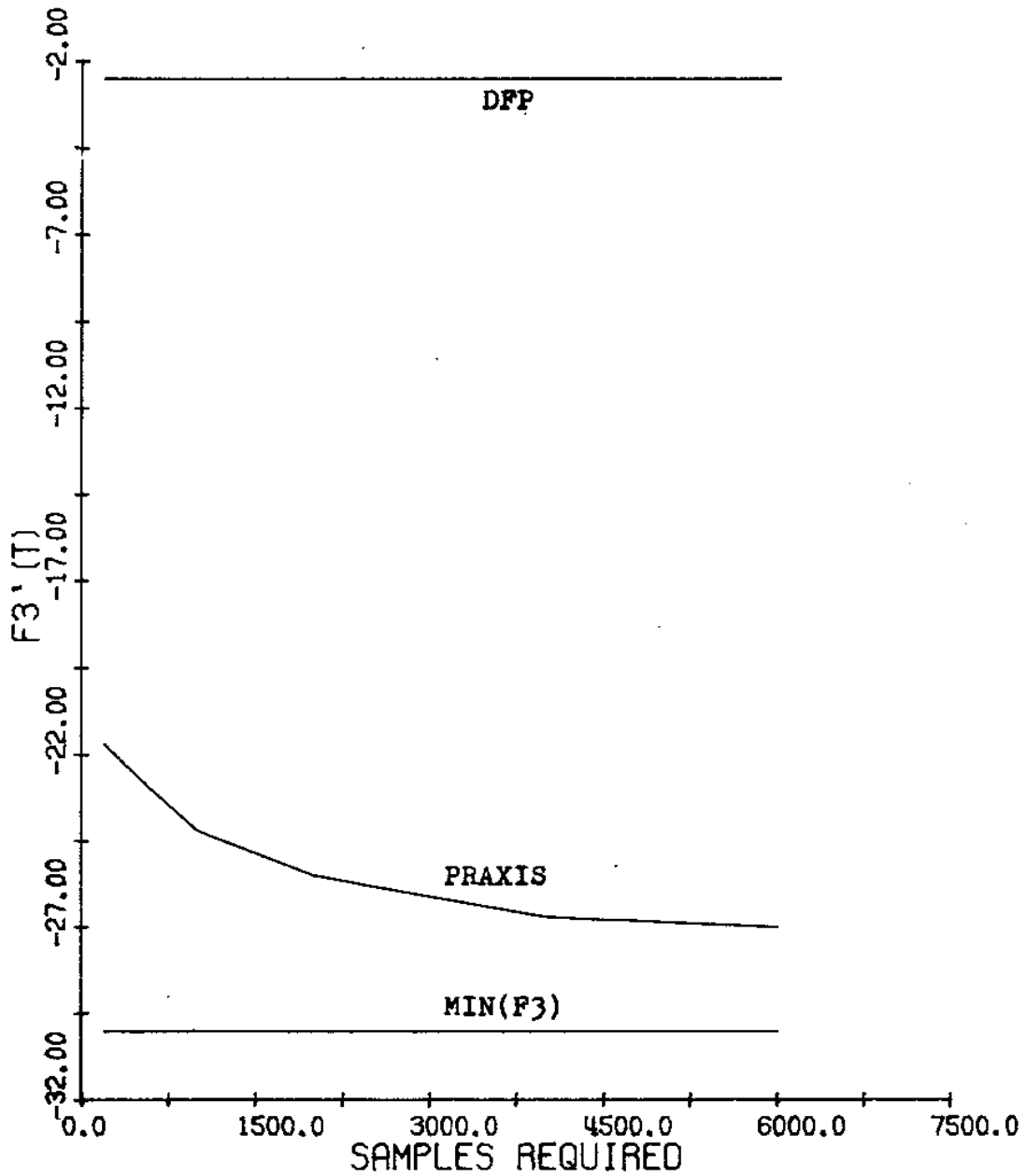


Figure 5.5: Off-line performance curves for PRAXIS and DFP in local mode on F3.

FIG. 5.6: ON-LINE PERFORMANCE ON F3

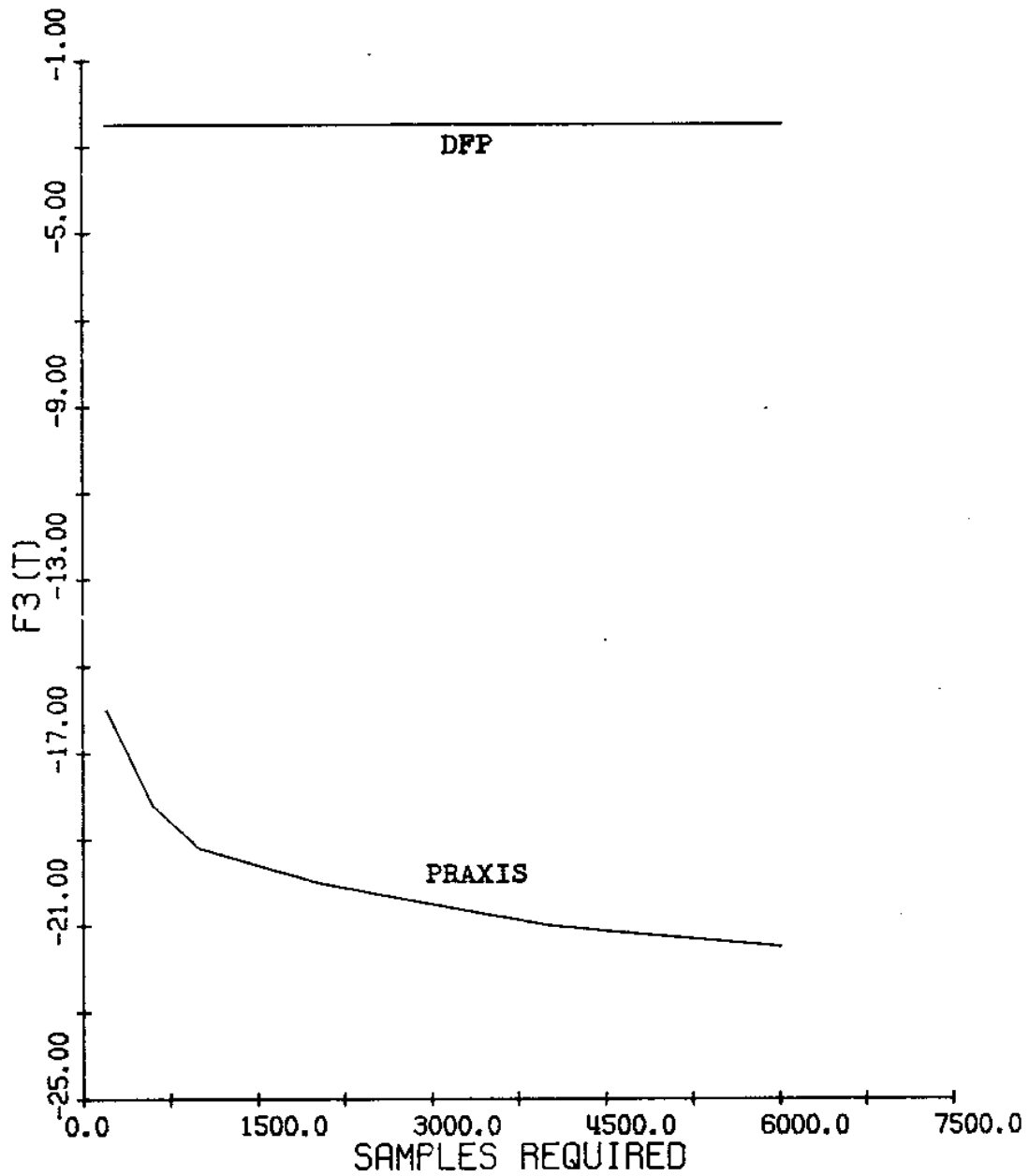


Figure 5.6: On-line performance curves for PRAXIS and DFP in local mode on F3.

curves generated on F_4 . Recall that F_4 was a high dimensional quartic with Gaussian noise. Here we see a considerable difference in the performance of the algorithms. PRAXIS seemed to make no better progress than random search on F_4 , while DFP easily outperformed the genetic algorithms. This suggests that PRAXIS is considerably more sensitive to noise than DFP. To verify this, PRAXIS was evaluated on F_4 with the Gaussian noise reduced from $N(0,1)$ to $N(0,.01)$. As illustrated, this resulted in considerable improvement in the performance of PRAXIS. It is interesting to speculate why PRAXIS is so much more sensitive to noise. Recall that PRAXIS constructs a conjugate direction without derivatives via a sequence of n one-dimensional minimizations. It is quite easy to imagine that this process is sensitive to noise, particularly with high-dimensional problems.

Finally, figures 5.9 and 5.10 illustrate the local performance curves generated on F_5 . The results are pretty much as expected. In local mode, both PRAXIS and DFP converge rapidly to the nearest local minimum, which when evaluated over a number of independent trials with random starting points yields convergence to the average value of F_5 on its local minima.

At this point it is fairly easy to predict what will happen when we switch PRAXIS and DFP to global mode. On F_1 and F_2 there will be essentially no change in off-line performance since convergence is already

FIG. 5.7: OFF-LINE PERFORMANCE ON F4

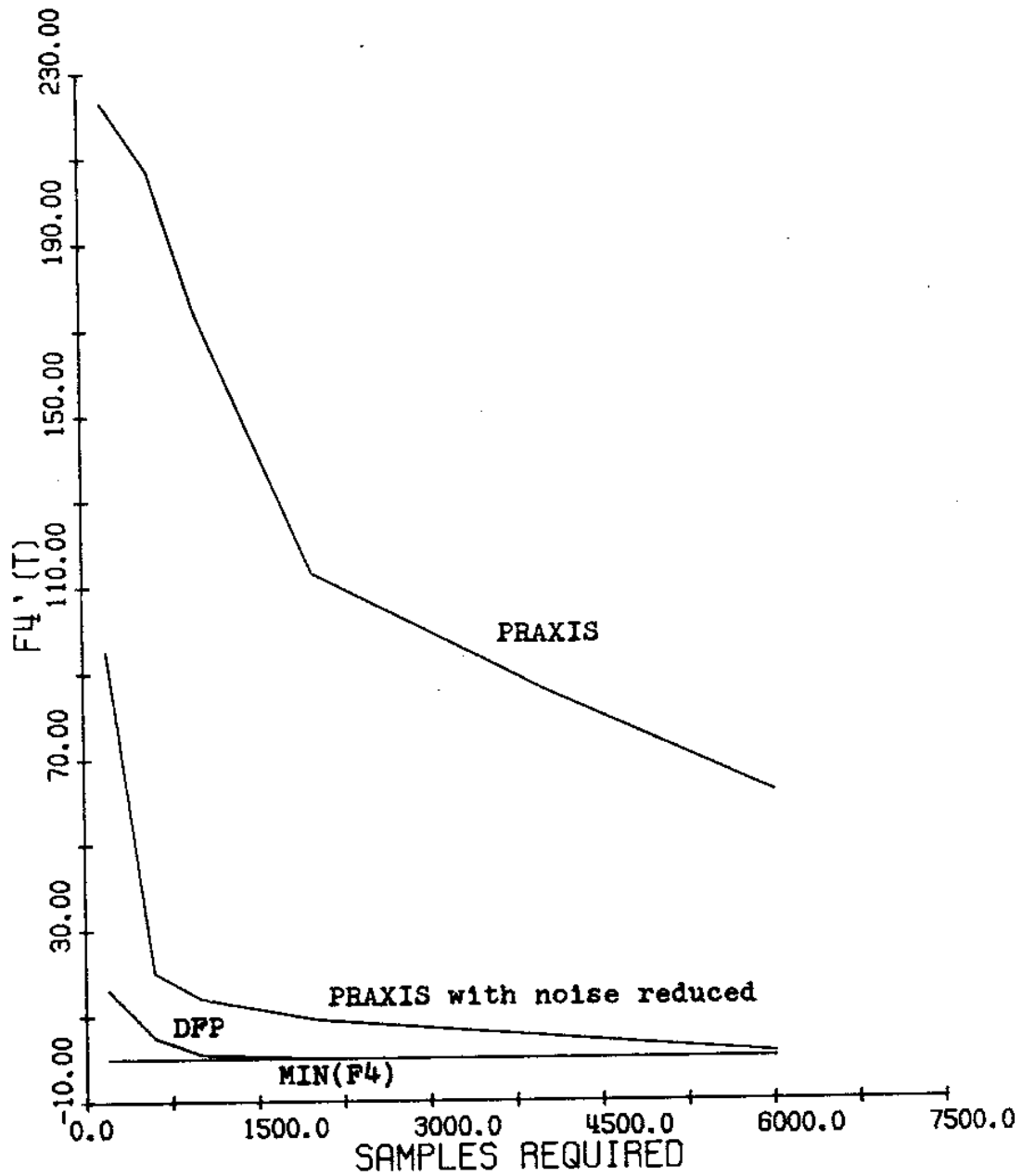


Figure 5.7: Off-line performance curves for PRAXIS and DFP in local mode on F4.

FIG. 5.8: ON-LINE PERFORMANCE ON F4

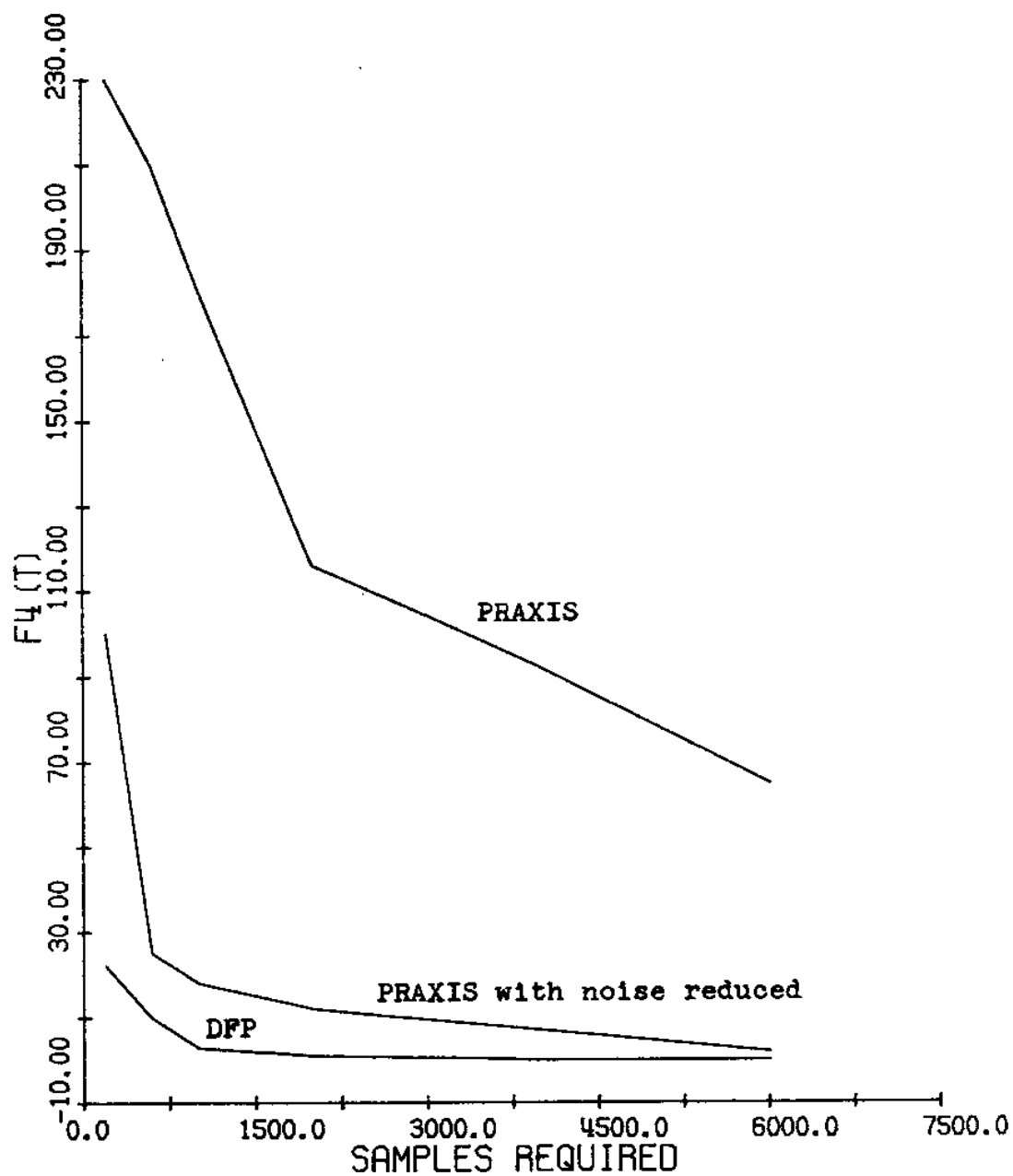


Figure 5.8: On-line performance curves for PRAXIS and DFP in local mode on F4.

FIG. 5.9: OFF-LINE PERFORMANCE ON F5

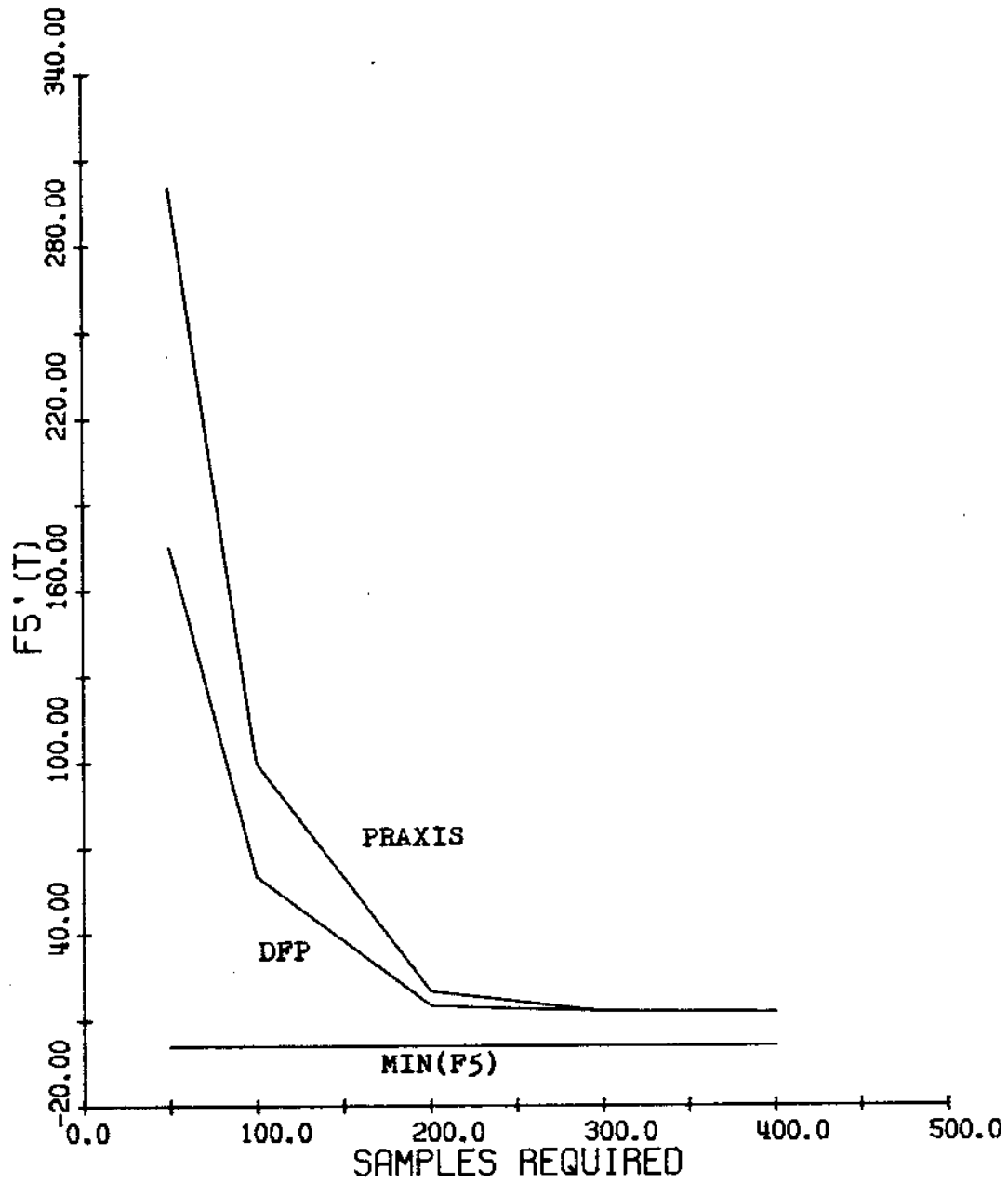


Figure 5.9: Off-line performance curves for PRAXIS and DFP in local mode on F5.

FIG. 5.10: ON-LINE PERFORMANCE ON F5

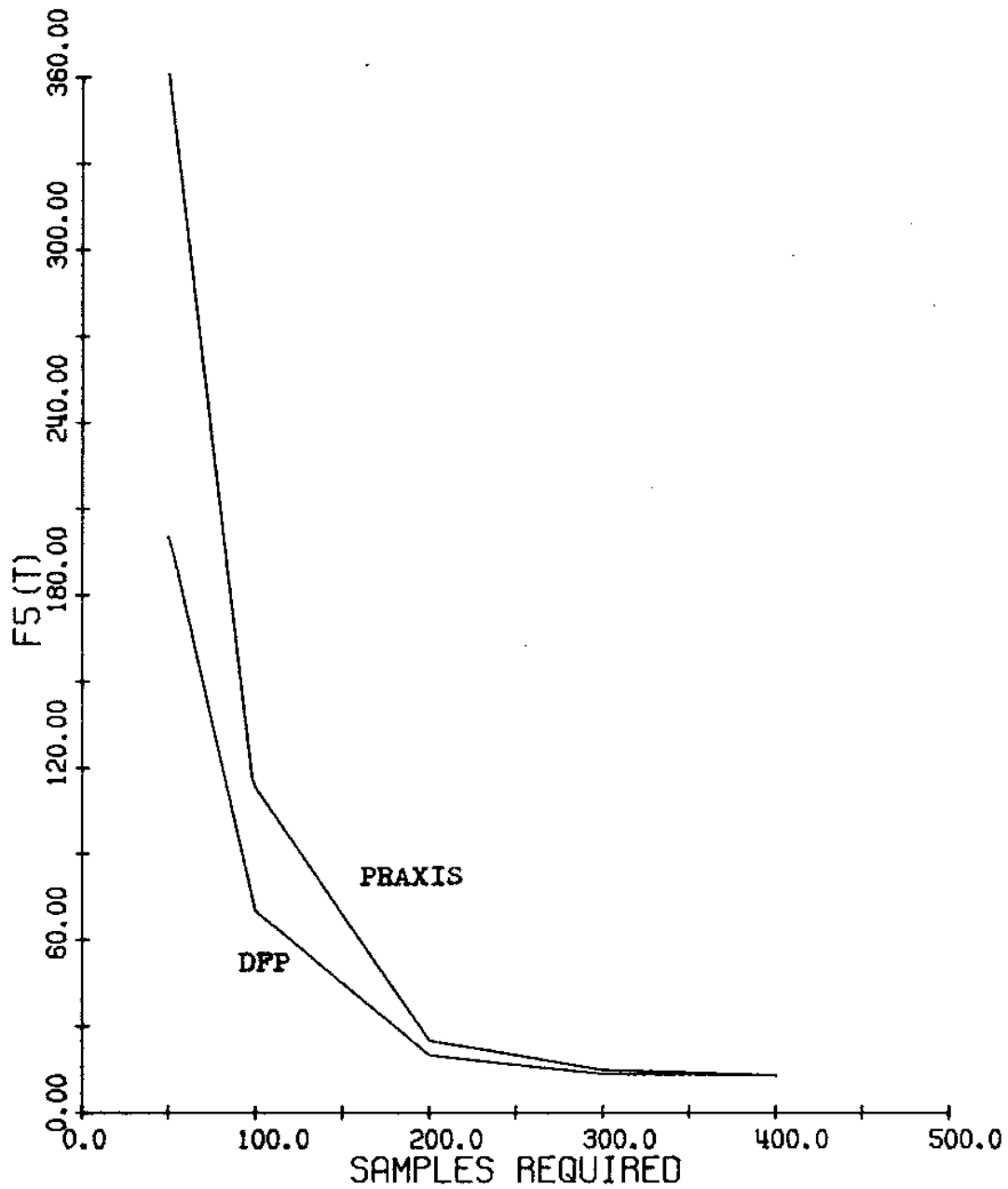


Figure 5.10: On-line performance curves for PRAXIS and DFP in local mode on F5.

achieved within 6000 trials. Notice, however, that re-starting a local optimizer will have a definite effect on on-line performance, degrading it considerably. This is the same kind of tradoff between local and global search we observed with the genetic algorithms.

Switching to global mode on F3 will not affect the performance of PRAXIS at all since it did not converge in local mode within 6000 trials. Global mode will, however, improve the performance of DFP on F3, but, as illustrated by 5.11, it can do no better than random search since each restart is followed almost immediately by convergence. As a consequence both are outperformed, for example, by R4 on F3.

On F4, PRAXIS in global mode is unchanged since it did not converge in 6000 trials. Since DFP did converge, switching to global mode will leave off-line performance unchanged, and degrade on-line performance.

The interesting case is, of course, the effect of switching PRAXIS and DFP to global mode on the performance curves for F5. Since in local mode both PRAXIS and DFP converged to the nearest local minimum in about 300-400 trials, each gets restarted about 15-20 times in 6000 trials. Since each of the 25 local optima is about the same size, we would expect that on the average the global minimum will not be found in 6000 trials. Figure 5.12 illustrates that this is, in fact, true for both optimizers, and indicates that R4, for

FIG 5.11: OFF-LINE PERFORMANCE ON F3

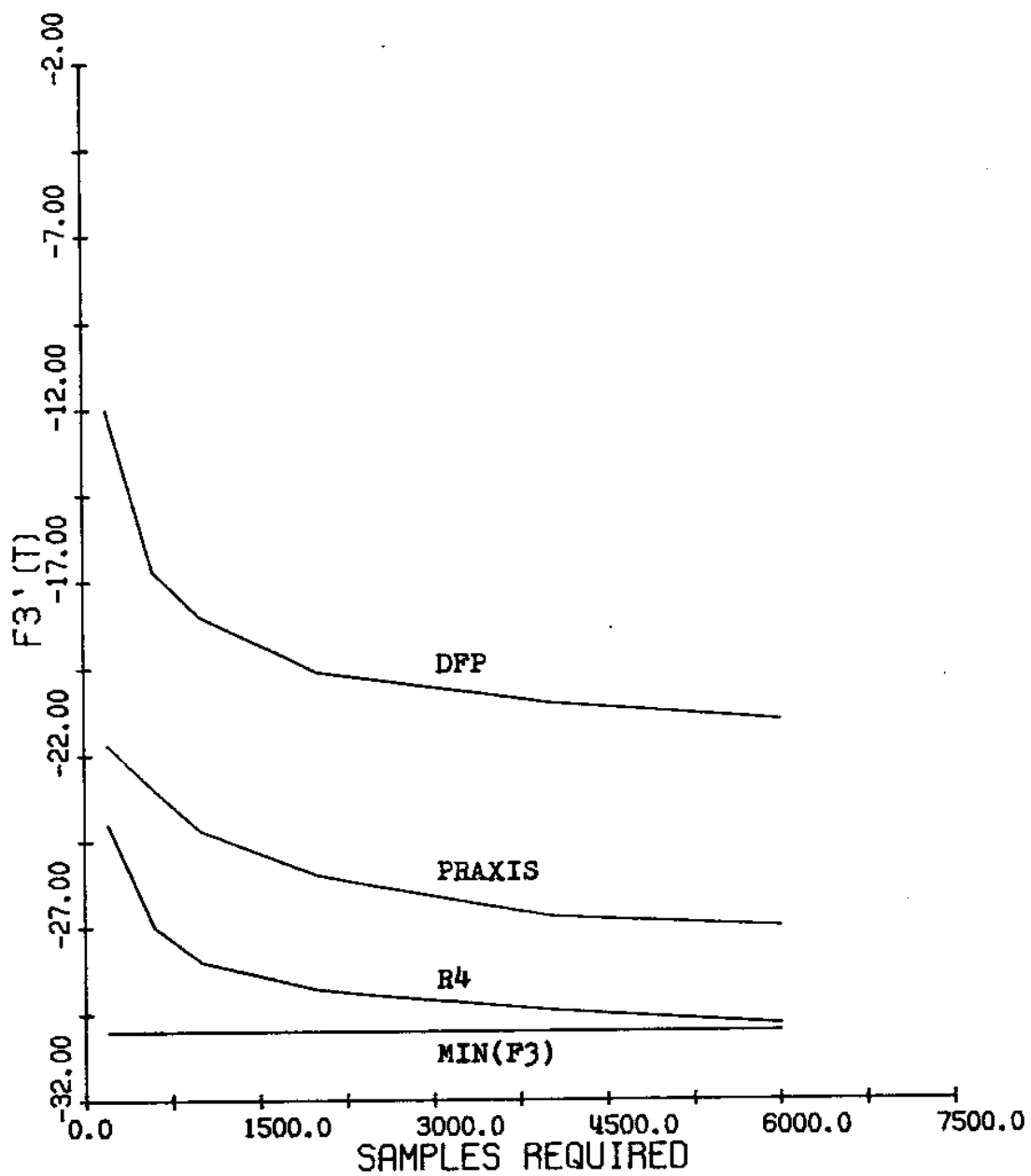


Figure 5.11: Off-line performance curves for PRAXIS and DFP in global mode on F3.

FIG. 5.12: OFF-LINE PERFORMANCE ON F5

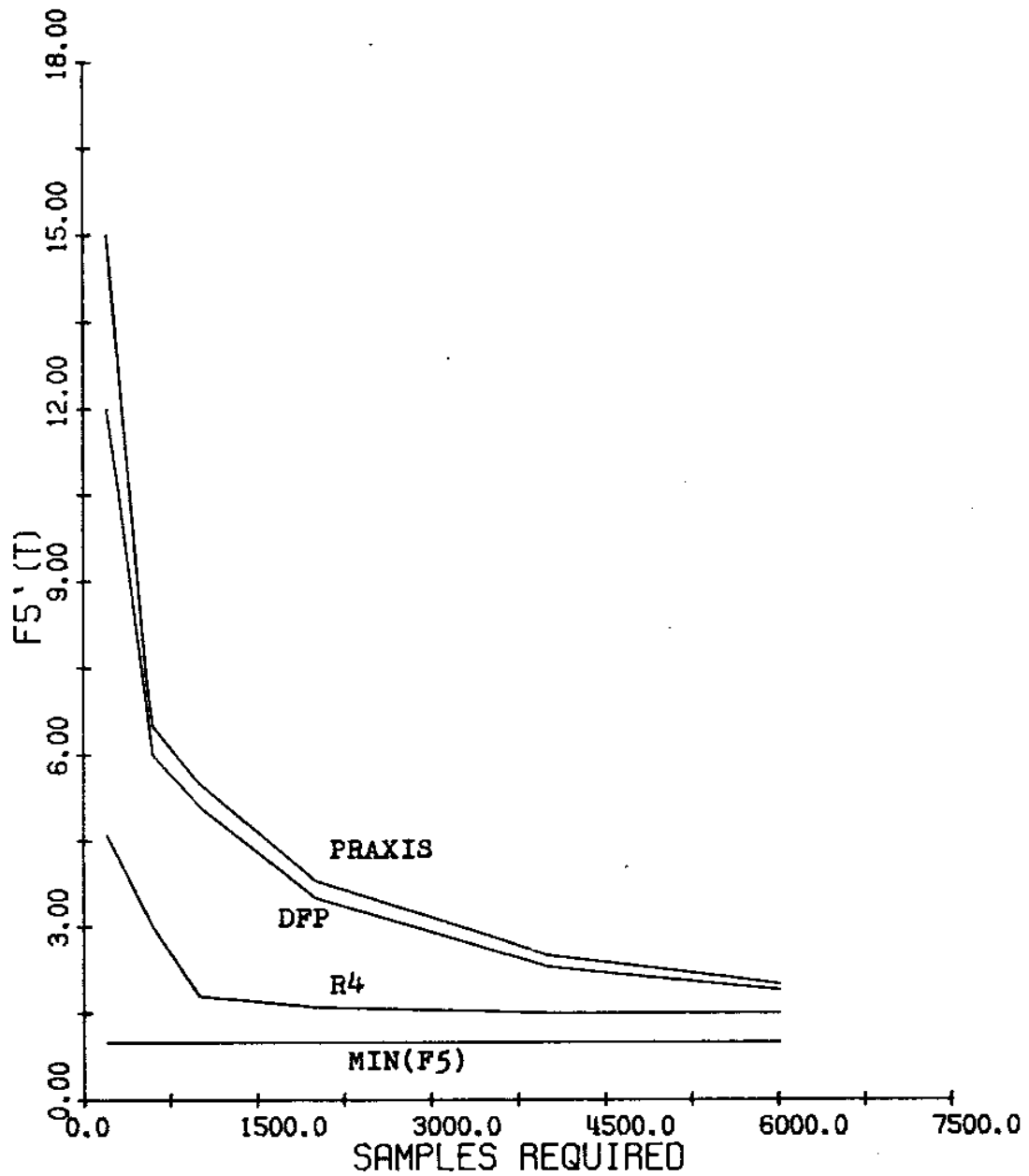


Figure 5.12: Off-line performance curves for PRAXIS and DFP in global mode on F5.

example, outperforms both on F5.

Finally, tables 5.1a and 5.1b give the off-line and on-line performance indices for PRAXIS and DFP evaluated over 6000 trials in both local and global mode. In general DFP outperformed PRAXIS on E, both in local and global mode. This is particularly true for on-line performance where the random search element in PRAXIS degraded performance considerably. Notice that both present a tradeoff between local and global mode. In local mode, rapid convergence to a possibly non-optimal minimum yields better on-line performance. On the other hand, restarting the optimizers increases the chances of finding the global optimum at the expense of on-line performance. In global mode, DFP came in a close second to R4 on off-line performance. DFP did better on F1, F2, and F4 while R4 was clearly the winner on F3 and F5. With respect to on-line performance, DFP outperformed R4 in local mode but not in global mode.

5.5 Summary

In this chapter we have attempted to provide a point of comparison for the behavior of genetic plans on E by evaluating the performance of two well-known function optimization techniques: a conjugate direction method and a variable metric method. Each were evaluated on E over 6000 trials in both local (normal) mode and a global mode which continued to restart the algorithms

T=6000	Local PRAXIS	Global PRAXIS	Local DFP	Global DFP	R4(50, .001, .6, 1.0)	Random Search
* $\mathcal{I}P_1(T)$.003	.003	.002	.002	.114	.36
* $\mathcal{I}P_2(T)$.051	.051	.003	.003	.221	.35
* $\mathcal{I}P_3(T)$	-25.47	-25.47	-2.5	-18.27	-28.2	-22.7
* $\mathcal{I}P_4(T)$	135.39	135.39	3.95	3.95	17.62	66.3
* $\mathcal{I}P_5(T)$	18.65	10.52	17.14	9.63	3.34	4.82
* $\mathcal{X}_E(T)$	25.72	24.1	3.72	-9.37	-1.38	9.83

Table 5.1a: Off-line performance indices for PRAXIS and DFP on E.

	Local PRAXIS	Global PRAXIS	Local DFP	GLOBAL DFP	$R_4(50, .001, .6, 1.0)$	Random Search
$T=6000$						
$x_{P1}(T)$.005	6.55	.004	4.71	2.32	26.2
$x_{P2}(T)$	1348.6	2967.6	.042	11.57	34.76	494.05
$x_{P3}(T)$	-16.84	-16.84	-2.5	-2.5	-26.49	-2.5
$x_{P4}(T)$	149.57	149.57	5.62	54.62	40.73	249.6
$x_{P5}(T)$	21.98	203.17	18.57	187.56	34.34	473.3
$x_E(T)$	244.2	662.01	4.35	51.19	17.12	147.61

Table 5.1b: On-line performance indices for PRAXIS and DFP on E.

after convergence if 6000 trials had not been allocated. In general, we found that the variable metric method, DFP, outperformed the conjugate direction method. In fairness to PRAXIS, however, we must remember that DFP was given exact derivative information about the test functions in E upon request. If the derivatives had been estimated or if a cost of more than n function evaluations had been exacted for each derivative computation, the differences in performance would have been less. During the evaluation, two interesting facts about PRAXIS were uncovered. In the first place, its heuristic strategies for avoiding stagnation on resolution ridges exacted a heavy toll when measuring on-line performance. This, of course, reflects the emphasis of the function optimization literature on convergence. Secondly, PRAXIS was seen to be quite sensitive to noise, performing no better on F^4 than random search. This suggests that the matrix updating techniques used by DFP to generate the conjugate directions of search are considerably less sensitive to noise than the constructive techniques used by PRAXIS, particularly in high-dimensional spaces.

Finally, we saw that DFP performed about as well as the genetic plans on E, but the tradeoffs were sharply drawn. On functions F_1 , F_2 , and F^4 which approximate the assumptions made by DFP about the function to be minimized, DFP was clearly the better choice. The

situation is, however, exactly reversed on F3 and F5 with R4 the obvious choice. These observations suggest that the genetic plans hold a valid position in both the fields of adaptation and function optimization, filling the gap between the efficient local search techniques and inefficient random search.

Chapter 6

SUMMARY AND CONCLUSIONS

We began this thesis by introducing a formalism for the study of adaptive systems and, within this framework, we defined a means of evaluating the performance of adaptive systems. The central feature of the evaluation process was the concept of robustness: the ability of an adaptive system to rapidly respond to its environment over a broad range of situations. To provide a concrete measure of robustness, a family E of environment response surfaces was carefully chosen to include representatives of a wide variety of response surfaces. When evaluating an adaptive system on a member of E , two distinct performance curves were monitored during adaptation: on-line performance and off-line performance. On-line performance evaluated every trial produced during adaptation, reflecting those situations in which an adaptive system is used to dynamically alter the performance of a system. Off-line performance evaluated only trials produced during adaptation which resulted in improved performance, reflecting situations in which testing can be done independently of the system being controlled.

Within this evaluation framework, a class of genetic adaptive systems was introduced for analysis and evaluation. These artificial genetic systems, called reproductive plans, generate adaptive responses by simulating

the information processing achieved in natural systems via the mechanisms of heredity and evolution. By introducing the concept of hyperplane partitions of the representation space, it was shown that reproductive plans have good theoretical properties with respect to the optimal allocation of trials to competing hyperplane partition elements. The performance of an elementary member, R1, of this class of genetic plans was evaluated on E and was shown to be superior to pure random search both in on-line and off-line performance. However, as defined, R1 was seen to converge quite regularly to a non-optimal plateau. Analysis suggested that this was due in part to the stochastic side-effects of random samples on a finite population. The effects of varying four parameters in the definition of R1 were analyzed in the hope of removing the problem of premature convergence and improving the performance of R1 on E. Increasing the population size was shown to reduce the stochastic effects and improve long-term performance at the expense of slower initial response. Increasing the mutation rate was seen to improve off-line performance at the expense of on-line performance. Reducing the crossover rate resulted in an overall improvement in performance, suggesting that producing a generation of completely new individuals was too high a sampling rate. Finally, reducing the generation gap was shown to yield the same kind of overall improvement in performance as crossover, but not as

dramatic.

As an alternative to modifying the parameters of R1, several modifications to the basic algorithm were analyzed for their effects on premature convergence and performance. An elitist policy favoring hyperplanes which produced the best individuals was shown to improve local search performance at the expense of global search. Modifying the sampling technique so that the actual number of offspring of an individual more closely approximated the expected value produced the best overall improvement in performance. Combining the expected value model with the elitist policy generated the best performance of any of the genetic plans analyzed on E, although the problem of premature convergence on F5 still remained. Both increasing the mutation rate and including a crowding factor were shown to resolve the problem of local convergence on R5, but at the expense of performance on the unimodal surfaces. Finally, a brief analysis of a generalized crossover model produced no significant improvement in the performance of genetic plans on E.

To provide an alternate point of comparison for performance on E (in addition to random search), two standard function optimization techniques were also evaluated on E. Each was run in their normal (local search) mode as well as a global mode in which, after local convergence, they were restarted at random starting points in an attempt to find the global minimum. Both

techniques outperformed the genetic algorithms on those surfaces in E for which they were designed. However, the genetic algorithms were seen to be superior on the discontinuous and multimodal surfaces in E .

As a consequence of these studies, several points of interest have been brought out. In the first place, it seems fairly clear that it is difficult, if not impossible, to simultaneously provide high-level off-line and on-line performance. Fortunately, most applications demand one, but not both. But it would be nice to provide a single solution to both. Off-line performance emphasizes convergence while on-line performance stresses short-term performance. As we have seen, better convergence properties are obtained by bold exploration in the early stages of adaptation, while short term performance is improved by the more conservative sampling policies. These distinctions are sharpened by the fact that in any practical situation evaluation is performed over a relatively short time interval. The longer the evaluation period, the less important short-term performance becomes.

A second point of interest brought out is the disparity between the mathematical characteristics of genetic plans and their implementation behavior. As we have seen, the fact that genetic plans support a finite population over a finite period of time can lead to considerably lower performance levels than predicted math-

ematically. Because the genetic plans operate in terms of a sequence of trial allocation decisions based on finite sampling distributions and sample means, they were seen to be quite sensitive to the associated stochastic errors. By minimizing as much as possible these stochastic side-effects, the implementation performance of genetic plans was improved considerably.

A third point of interest was brought out in the comparative analysis of function optimization techniques. As with many other difficult problems in computer science, adaptation poses the tradeoff between a general solution to the problem and performance. By making assumptions about the kind of response surfaces to be faced, extremely efficient performance can be generated on a relatively small class of functions. It is clear that no genetic algorithm is going to minimize a convex quadratic function in 50 trials. On the other hand, it is clear that if such assumptions are not met, then genetic algorithms provide a considerably better alternative than random search without making any assumptions about the form of the response surface.

These studies also suggest several interesting areas for future research. Neither time nor resources permitted extensive evaluation of the generalized crossover operator. Further study may support the intuition that performance improvements could be achieved for small increases in the number of crossover points.

Another area worth exploring is to consider the effects of a diploid representation. That is, each gene position has two alleles with a "dominance" map specifying its functional value. Dominance has a very direct bearing on the problem of allele loss in finite populations.

Finally, it would be of interest to explore the possibility of introducing "species" into the genetic algorithm. This also has direct bearing on the problem of allele loss allowing, for example, exponential exploitation of a local minimum without the problem of complete population dominance.